1D Barcode VCL Components

User Manual

Version: 6.0.1.2015

Copyright © 2001-2015 Han-soft Corporation. All rights reserved.

2015-09-08

Table of Contents

Table of Contents 1
Chapter 1. Introduction 5
1.1 Overview 5
1.2 Main features 5
Chapter 2. Installation 6
2.1 Trial user 6
2.1 Registered user 6
Chapter 3. Quick start 8
3.1 How to use the barcode components 8
3.2 How to use the barcode components with a database 8
3.3 How to use the barcode components with QuickReport 9
3.4 How to use the barcode components with FastReport 9
3.5 How to use the barcode components with ReportBuilder 9
3.6 How to use the barcode components with ACE Reporte 10
3.7 How to use the barcode components with Rave Reports 10
3.8 How to print a barcode symbol to paper 11
3.9 How to save a barcode symbol to picture file 12
3.10 How to encode the UNICODE text in a Code128/EAN128 symbol 12
Chapter 4. Reference 14
4.1 TBarcode1D 14
4.1.1 TBarcode1D_AP4SC 14
4.1.2 TBarcode1D_BC309 16
4.1.3 TBarcode1D_BC412 17
4.1.4 TBarcode1D_Channel 18
4.1.5 TBarcode1D_Clocked35 19
4.1.6 TBarcode1D_Codabar 20
4.1.7 TBarcode1D_Code11 21
4.1.8 TBarcode1D_Code128 22
4.1.9 TBarcode1D_Code25Datalogic 23
4.1.10 TBarcode1D_Code25Industrial 24
4.1.11 TBarcode1D_Code25Interleaved 25
4.1.12 TBarcode1D_Coed25Invert 26
4.1.13 TBarcode1D_Code25Matrix 27
4.1.14 TBarcode1D_Code32 28
4.1.15 TBarcode1D_Code39 29
4.1.16 TBarcode1D_Code39Ext 30
4.1.17 TBarcode1D_Code93 31
4.1.18 TBarcode1D_Code93Ext 32
4.1.19 TBarcode1D_Coop25 33
4.1.20 TBarcode1D_CPCBin 34
4.1.21 TBarcode1D_EAN128 35
4.1.22 TBarcode1D_EAN13 36
4.1.23 TBarcode1D_EAN2 37
4.1.24 TBarcode1D_EAN5 38
4.1.25 TBarcode1D_EAN8 39
4.1.26 TBarcode1D_FIM 40
4.1.27 TBarcode1D_Flattermarken 41
4.1.28 TBarcode1D_IATA 42
4.1.29 TBarcode1D_Identcode 43
4.1.30 TBarcode1D_ITF14 44

4.1.31 TBarcode1D_ITF16 45
4.1.32 TBarcode1D_ITF6 46
4.1.33 TBarcode1D_JapanPost 47
4.1.34 TBarcode1D_KIX4S 48
4.1.35 TBarcode1D_Leitcode 49
4.1.36 TBarcode1D_MSI 50
4.1.37 TBarcode1D_OneCode 51
4.1.38 TBarcode1D_Patch 52
4.1.39 TBarcode1D_PharmacodeOneTrack 53
4.1.40 TBarcode1D_PharmacodeTwoTrack 54
4.1.41 TBarcode1D_Planet 55
4.1.42 TBarcode1D_Plessey 56
4.1.43 TBarcode1D_PostBar 57
4.1.44 TBarcode1D_PostNet 59
4.1.45 TBarcode1D_PZN 59
4.1.46 TBarcode1D_RM4SCC 60
4.1.47 TBarcode1D_Telepen 61
4.1.48 TBarcode1D_UPCA 62
4.1.49 TBarcode1D_UPCE 63
4.1.50 TBarcode1D_UPCE0 64
4.1.51 TBarcode1D_UPCE1 65
4.1.52 TBarcode1D_UPU 66
4.2 TBarcodeCustomParameters 67
4.3 TBarcodeTextDefine 74
4.4 TDBBarcode1D 77
Chapter 5. FAQs 78
5.1 How to download the full version 78
5.2 I forgot my license key 78
Chapter 6. Information 79
6.1 Support 79
6.2 Purchase 79
6.3 Contact us 79
6.4 End-User License Agreement 80
Annex A. Properties 82
A.1 TBarcode1D 82
A.1.1 AutoCheckDigit 82
A.1.2 Barcode 82
A.1.3 BarcodeHeight 82
A.1.4 BarcodeWidth 83
A.1.5 BarColor 84
A.1.6 BarColors (TBarcode1D_PharmacodeOneTrack, TBarcode1D_PharmacodeTwoTrack) 84
A.1.7 BearerBars (TBarcode1D_ITF6, TBarcode1D_ITF14, TBarcode1D_ITF16) 84
A.1.8 BearerWidth (TBarcode1D_ITF6, TBarcode1D_ITF14, TBarcode1D_ITF16) 85
A.1.9 Bidirectional (TBarcode1D_Plessey) 85
A.1.10 Channel (TBarcode1D_Channel) 85
A.1.11 CheckLength (TBarcode1D_EAN128) 86
A.1.12 CheckMethod (TBarcode1D_MSI) 86
A.1.13 CheckStart (TBarcode1D_EAN128) 86
A.1.14 CurrentChannel (TBarcode1D_Channel) 87
A.1.15 CurrentSubSet (TBarcode1D_Code128, TBarcode1D_EAN128) 87
A.1.16 Data 87

A.1.17 DisplayText 88
A.1.18 EncodeMode (TBarcode1D_Code128, TBarcode1D_EAN128) 88
A.1.19 ExtraChar (TBarcode1D_Telepen) 89
A.1.20 ExtraFontSize (TBarcode1D_UPCA, TBarcode1D_UPCE, etc.) 89
A.1.21 FIMType (TBarcode1D_FIM) 90
A.1.22 FullEncoded 90
A.1.23 Guards (TBarcode1D_Telepen) 90
A.1.24 Height 91
A.1.25 HideCheckDigitsText (TBarcode1D_PostBar) 91
A.1.26 Image 91
A.1.27 InitialSubSet (TBarcode1D_Code128, TBarcode1D_EAN128) 92
A.1.28 InterGap (TBarcode1D_PostBar, TBarcode1D_AP4SC, TBarcode1D_ITF14, etc.) 92
A.1.29 LeftMargin 92
A.1.30 LeftQuietZoneSpacing (TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_EAN13, etc.) 94
A.1.31 LeftSpacing (TBarcode_ITF6, TBarcode_ITF14, TBarcode_ITF16) 95
A.1.32 Link2D (TBarcode1D_EAN128) 95
A.1.33 Locked 95
A.1.34 Mod11Weighting (TBarcode1D_MSI) 96
A.1.35 Mode (TBarcode1D_Telepen) 96
A.1.36 Module 96
A.1.37 NumberCheckDigit (TBarcode1D_Code11) 97
A.1.38 NumCustomerInfo (TBarcode1D_AP4SC) 97
A.1.39 OddEncode (TBarcode1D_Telepen) 97
A.1.40 Orientation 98
A.1.41 Padding (TBarcode1D_Code25Interleaved) 98
A.1.42 PatchType (TBarcode1D_Patch) 98
A.1.43 Ratio 99
A.1.44 RightQuietZoneSpacing (TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_EAN13, etc.) 99
A.1.45 RightSpacing (TBarcode1D_ITF6, TBarcode1D_ITF14, TBarcode1D_ITF16) 100
A.1.46 Routing (TBarcode1D_OneCode) 100
A.1.47 ShortFinder (TBarcode1D_Channel) 100
A.1.48 ShowGuards (TBarcode1D_Code39, TBarcode1D_UPU, TBarcode1D_Code32, etc.) 100
A.1.49 ShowQuietZoneMark (TBarcode1D_ENA5, TBarcode1D_EAN8, TBarcode1D_EAN13, etc.) 101
A.1.50 SpaceColor 101
A.1.51 SplitText (TBarcode1D_PostBar, TBarcode1D_AP4SC, TBarcode1D_ITF14, etc.) 101
A.1.52 SplitText (TBarcode1D_OneCode) 101
A.1.53 StartCode (TBarcode1D_Codabar) 102
A.1.54 StopCode (TBarcode1D_Codabar) 102
A.1.55 Stretch 102
A.1.56 StretchTextHeight 103
A.1.57 TextAlignment 103
A.1.58 TextCSpacing 105
A.1.59 TextFont 105
A.1.60 TextHSpacing 106
A.1.61 TextPosition 106
A.1.62 TextVSpacing 107
A.1.63 TopMargin 107
A.1.64 Tracking (TBarcode1D_OneCode) 109
A.1.65 UKMode (TBarcode1D_Plessey) 109
TDBBarcode1D 110
A.2.1 Barcode1D 110
A.2.2 BindProperty 110

A.2.3 DataField 111
A.2.4 DataSource 111
A.2.5 Field 111
A.2.6 ReadOnly 111
Annex B. Methods 113
B.1 TBarcode1D 113
B.1.1 Assign 113
B.1.2 Clear 113
B.1.3 CopyToClipboard 113
B.1.3.1 CopyToClipboard - Syntax 1 114
B.1.3.2 CopyToClipboard - Syntax 2 115
B.1.3.3 CopyToClipboard - Syntax 3 116
B.1.4 Create 117
B.1.5 Destroy 118
B.1.6 Draw 118
B.1.7 DrawTo 118
B.1.7.1 DrawTo - Syntax 1 119
B.1.7.2 DrawTo - Syntax 2 120
B.1.7.3 DrawTo - Syntax 3 122
B.1.8 DrawToSize 124
B.1.8.1 DrawToSize - Syntax 1 124
B.1.8.2 DrawToSize - Syntax 2 126
B.1.8.3 DrawToSize - Syntax 3 128
B.1.9 Print 130
B.1.9.1 Print - Syntax 1 130
B.1.9.2 Print - Syntax 2 132
B.1.9.3 Print - Syntax 3 134
B.1.10 PrintSize 136
B.1.10.1 PrintSize - Syntax 1 136
B.1.10.2 PrintSize - Syntax 2 138
B.1.10.3 PrintSize - Syntax 3 141
B.1.11 Size 143
Annex C. Events 145
C.1 TBarcode1D 145
C.1.1 OnChange 145
C.1.2 OnDecodeText (TBarcode1D_Code128, TBarcode1D_EAN128) 145
C.1.3 OnDrawBarcode 146
C.1.4 OnDrawText 146
C.1.5 OnEncode (TBarcode1D_Code128, TBarcode1D_EAN128) 148
C.1.6 OnInvalidChar 148
C.1.7 OnInvalidDataChar 149
C.1.8 OnInvalidDataLength 149
C.1.9 OnInvalidLength 150

Chapter 1. Introduction

1.1 Overview

1D Barcode VCL Components is the most flexible and powerful VCL components package which lets you to easily add advanced barcode generation and printing features to your application.

1D Barcode VCL Components supports most popular Linear (1D), Clocked (1D), Postal Symbologies/Standards all-in-one solution including Codabar, Code 11, Code 25, Code 32, Code 39, Code 93, Code 128, EAN 2, EAN 5, EAN 8, EAN 13, EAN 128, UPC-A, UPC-E, ITF, Postal (USPS, British Royal Mail, Australia Post, KIX4S, DHL, etc), Telepen, Plessey, MSI and many more barcode standards.

The components package can be used together with 2D Barcode VLC Components package to create the EAN.UCC composite barcode symbols.

All morden versions of Delphi and C++ Builder are supported, including the Delphi 3, 4, 5, 6, 7, 2005, 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle and C++ Builder 4, 5, 6, 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle.

For Delphi XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle and C++ Builder XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle, both 32-bit and 64-bit components are included.

1D Barcode VCL Components are easy to use. Developers use them like any other VCL component.

1.2 Main features

- Allows to draw the barcode symbol to canvas (with scaling and rotating).
- Allows to print the barcode symbol to paper (with scaling and rotating).
- Most popular barcode symbologies are supported.
- Check digit is automatically calculated and added.
- All of the Quick Report, Fast Report, Report Builder, Rave Reports, and ACE Reporter are supported.
- The database function is supported, including the data access components (such as the FireDAC, dbExpress, BDE) and the LiveBindings.
- The Delphi 3, 4, 5, 6, 7, 2005, 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle, and C++ Builder 4, 5, 6, 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle are supported.
- It is visible in design time.
- For Delphi XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle, and C++ Builder XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle, both 32-bit and 64-bit components are included.
- Ability to scale and rotate the barcode symbols.
- Ability to create the EAN.UCC composite barcode symbols together with 2D Barcode VCL Components package.
- Optional human readable text is supported.
- All windows fonts can be used for the optional human readable text, its fore- and background colors can be changed freely.
- Fore- and background colors of barcode symbol can be changed freely.
- It's easy to use, and it has the excellent functionality.
- It's a very popular barcode components package.

Chapter 2. Installation

2.1 Trial user

Installation step by step:

1. Before installing the components package, please close all Delphi and C++ Builder IDEs.

   Note, for each IDE, if it's a clean installation, please run it at least once before installing the components package, then closes it and continues installation.

2. Run the installation file **barcode1d_tri.exe**, and then click on the "Next" button in the installation dialog box.
3. Read the **End-User License Agreement**. You must accept the terms of this agreement before continuing with the installation. And then click on the "Next" button.
4. Specify a target folder (it will be created if does not exist), the components package will be installed into it. And then click on the "Next" button.
5. All supported Delphi and C++ Builder IDEs will be listed automatically according on the existing IDEs in your computer. Please select the IDEs you want to install to them. And then click on the "Next" button.

   Note, The Delphi 3, 4, 5, 6, 7, 2005, 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle and C++ Builder 4, 5, 6, 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle are supported now. The Delphi 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle and C++ Builder 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle are listed as RAD Studio (Delphi and C++ Builder) 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle.

6. Specify a shortcuts folder in "Start Menu" for the components package. And then click on the "Next" button.Later, you can open the manual or remove the components package in the shortcuts folder.
7. Click on the "Install" button to complete the components package installation.
8. Click on the "Finish" button to close the installation dialog box.
9. You can start your IDE to use the components package now.

Note:

- If multi-user accounts want to use the components package, please install it into **the same folder** in each user session.
- For Delphi XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle and C++ Builder XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle, both 32-bit and 64-bit components are included.

2.1 Registered user

Installation step by step:

1. Before installing the components package, please close all Delphi and C++ Builder IDEs.

   Note, for each IDE, if it's a clean installation, please run it at least once before installing the components package, then closes it and continues installation.

2. Please uninstall the trial release using the "Uninstall" shortcuts in the "Start Menu" if it is installed in your computer.
3. Please download the installation package using the download link that's sent from us after you purchase the components package.If your download link doesn't work, please visit the "Manage your licenses" page to request a new download link. Please open the page then enter your order ID, license user name or license e-mail address to locate your order, then click on the oder ID to display it, choose a license and click on the "**Request a new download link**", the new download link will be sent to this license e-mail address automatically.
4. Run the installation file **barcode1d_ful.exe**, and then click on the "Next" button in the installation dialog box.
5. Read the **End-User License Agreement**. You must accept the terms of this agreement before continuing with the installation. And then click on the "Next" button.
6. Type your email address and the license key that they are sent from us after you purchase the components package, they are not case-sensitive. And then click on the "Next" button. If you forgot the license key, please visit the "Manage your licenses" page to retrieve it. Please open the page then enter your order ID, license user name or license e-mail address to locate your order, then click on the oder ID to display it, choose a license and click on the "**Retrieve the license key**", the license key will be sent to the license e-mail address automatically.
7. Specify a target folder (it will be created if does not exist), the components package will be installed into it. And then click on the "Next" button.
8. All supported Delphi and C++ Builder IDEs will be listed automatically according on the existing IDEs in your computer. Please select the IDEs you want to install to them. And then click on the "Next" button.

   Note, The Delphi 3, 4, 5, 6, 7, 2005, 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle and C++ Builder 4, 5, 6, 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle are supported now. The Delphi 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle and C++ Builder 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle are listed as RAD Studio (Delphi and C++ Builder) 2006, 2007, 2009, 2010, XE, XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle.

9. Specify a shortcuts folder in "Start Menu" for the components package. And then click on the "Next" button.Later, you can open the manual or remove the components package in the shortcuts folder.
10. Click on the "Install" button to complete the components package installation.
11. Click on the "Finish" button to close the installation dialog box.

12. You can start your IDE to use the components package now.

Note:

- If multi-user accounts want to use the components package, please install it into **the same folder** in each user session.
- After installation, please delete all ".dcu" files in your projects that they are built using the trial release of the components package, then re-build these projects.
- For Delphi XE2, XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle, and C++ Builder XE3, XE4, XE5, XE6, XE7, XE8, 10 Seattle, both 32-bit and 64-bit components are included.

Chapter 3. Quick start

3.1 How to use the barcode components

Usage:

1. Put a TBarcode1D barcode component, such as the TBarcode1D_Code39, TBarcode1D_EAN13, and TBarcode1D_Code128 to your form.
2. Put a TImage control to your form.
3. Set the Image property of the barcode component to the TImage control.

   You can link single TImage control to multiple TBarcode1D components in order to display multiple barcode symbols in the TImage control (using the LeftMargin and TopMargin properties to specify the position for every barcode symbol).

Note:

If the barcode symbol isn't displayed, or it is wrong, please check whether the length of barcode text exceeds the maximum length limit, or whether there is any invalid character in the barcode text.

You can create the OnInvalidLength and OnInvalidDataLength (only for Delphi/C++ Builder 2009 or later) event handles to catch the invalid barcode length exception. And create the OnInvalidChar and OnInvalidDataChar (only for Delphi/C++ Builder 2009 or later) event handles to catch the invalid character in the barcode text.

Also, please check whether the TImage control is large enough to accommodate entire barcode symbol.

3.2 How to use the barcode components with a database

Use the classic data access components such as BDE, dbExpress, FireDAC, AnyDAC, etc.

1. Put a TBarcode1D barcode component, such as the TBarcode1D_Code39, TBarcode1D_EAN13, and TBarcode1D_Code128 to your form.
2. Put a TDBBarcode1D component to your form.
3. Set the Barcode1D property of the TDBBarcode1D component to your TBarcode1D barcode component.
4. Set the DataSource property of the TDBBarcode1D component to your TDataSource component.
5. Set the DataField property of the TDBBarcode1D component to a field in your dataset.
6. If you use the Delphi/C++ Builder 2009 or later, Set the BindProperty property of the TDBBarcode1D component to indicate which property of the TBarcode1D component the data field value will be applied to.
7. If you want to represent the barcode symbol in form or QuickReportreport, put a TImage control to your form, or put a TQRImage or TQRGzImage control to your report. Set the Image property of the TBarcode1D barcode component to the TImage, TQRImage, or TQRGzImage control. The data in the data field will be represented as a barcode symbol in your TImage, TQRImage, or TQRGzImage control.
8. You can use the Print method of the TBarcode1D component to print the barcode symbol to paper, or use the DrawTo method of the TBarcode1D component to draw the barcode symbol to any TCanvas.
9. Also, you can use the SaveToFile method of the TImage control that is linked to the TBarcode1D barcode component to save the barcode symbol as a picture file.

Note, You can bind multiple TDBBarcode1D and TBarcode1D component pairs to a data field in order to represent the data field with multiple barcode symbols.

Use the LiveBindings (Delphi/C++ Builder XE2 or later).

1. Put a TBarcode1D barcode component, such as the TBarcode1D_Code39, TBarcode1D_EAN13, and TBarcode1D_Code128 to your form.
2. Open the "LiveBindings Designer", click on the barcode component inthe form to select it, change the "Visible Element" sub-item of the "LiveBindings Designer" item to true in the "Object Inspector".
3. Right-click on the barcode component in the "LiveBindingsDesigner", execute the "Bindable Members..." menu item, check the"Barcode" property or the "Data" property (only for the Delphi/C++ Builder 2009 or later) in the "Bindable Members" dialog, click on the "OK" button to close it.

   Note, for the TBarcode1D_FIM component, please check the "FIMType" or the "Data" (only for theDelphi/C++ Builder 2009 or later) property. For the TBarcode1D_Patch component, please check the "PatchType" or the "Data" (only for the Delphi/C++ Builder 2009 orlater) property. For the TBarcode1D_OneCode component, please check the "Tracking" and "Routing" properties or the "Data" property (only for the Delphi/C++ Builder 2009 or later).
4. Link the Barcode property or the Data property (only for the Delphi/C++ Builder2009 or later) of the TBarcode1D barcode component to your data field in the TBindSourceDB component, or a string property of other control.

   Note, for the TBarcode1D_FIM component, please link the FIMType or the Data (only for the Delphi/C++ Builder 2009 orlater) property. For the TBarcode1D_Patch component, please link the PatchType or the Data (only for the Delphi/C++ Builder 2009or later) property. For the TBarcode1D_OneCode component, please link the Tracking and Routing properties or the Data property (only for the Delphi/C++ Builder 2009 or later).

5. If you want to represent the barcode symbol in form or QuickReportreport, put a TImage control to your form, or put a TQRImage or TQRGzImage control to your report. Set the Image property of the TBarcode1D barcode component to the TImage, TQRImage, or TQRGzImage control. The data in the data field will be represented as a barcode symbol in your TImage, TQRImage, or TQRGzImage control.
6. You can use the Print method of the TBarcode1D component to print the barcode symbol to paper, or use the DrawTo method of the TBarcode1D component to draw the barcode symbol to any TCanvas.
7. Also, you can use the SaveToFile method of the TImage control that is linked to the TBarcode1D barcode component to save the barcode symbol as a picture file.

Note, You can link multiple TBarcode1D components to a data field in order to represent the data field with multiple barcode symbols.

3.3 How to use the barcode components with QuickReport

Usage:

1. Put a TBarcode1D barcode component, such as the TBarcode1D_Code39, TBarcode1D_EAN13, and TBarcode1D_Code128 to your form.

   Also, put a TDBBarcode1D component to the form and link the TBarcode1D component to the TDBBarcode1D component if the database support is required.
2. Put a TQRImage or TQRGzImage control to your report.
3. Set the Image property of the barcode component to the TQRImage or TQRGzImage control.

   You can link single TQRImage or TQRGzImage control to multipleTBarcode1D components in order to display multiple barcode symbols in the TQRImage or TQRGzImage control (using the LeftMargin and TopMargin properties to specify the position for every barcode symbol).

Note:

If the barcode symbol cannot be read, please don't reduce/stretch width of the barcode symbol (set the Stretch property to false). You can change the barcode symbol width by changing its Module property value.

Also, please check whether the TQRImage or TQRGzImage control is large enough to accommodate entire barcode symbol.

3.4 How to use the barcode components with FastReport

Usage:

1. Edit your report, put a TfrxPictureView control to your report in order to present the barcode symbol.
2. Put a TBarcode1D barcode component, such as the TBarcode1D_Code39, TBarcode1D_EAN13, and TBarcode1D_Code128 to your form that the TfrxReport component is in it.

   Also, put a TDBBarcode1D component to the form and link the TBarcode1D component to the TDBBarcode1D component if the database support is required.
3. Create the OnBeforePrint event function for the TfrxReport component.

   In the event function, change the properties of the TBarcode1D component such as Barcode, Module, and Ratio, and adjust the bitmap size of the TfrxPictureView control in order to accommodate entire barcode symbol, then use the DrawTo method of the TBarcode1D component to draw the barcode symbol to the TfrxPictureView control.

   For example:

```
var
  BarcodeWidth, BarcodeHeight, SymbolWidth, SymbolHeight: Integer;
begin
  ......
  Barcode1D_Code391.Barcode := '1235678';
  Barcode1D_Code391.Module := 2;
  ......
  with  TfrxPictureView(frxReport1.FindObject('Picture1')).Picture.Bitmap do
  begin
    Barcode1D_Code391.DrawToSize(BarcodeWidth, BarcodeHeight, SymbolWidth, SymbolHeight, Canvas);
    Width := BarcodeWidth;
    Height := 100;
    Barcode1D_Code391.DrawTo(Canvas, 0, 0);
  end;
```

   Note, if you use old FastReport 2.x, please use the OnBeginBand event.

3.5 How to use the barcode components with ReportBuilder

Usage:

1. Edit your report, put a TppImage control to your report in order to present the barcode symbol.

2. Put a TBarcode1D barcode component, such as the TBarcode1D_Code39, TBarcode1D_EAN13, and TBarcode1D_Code128 to your form that the TppReport component is in it.

   Also, put a TDBBarcode1D component to the form and link the TBarcode1D component to the TDBBarcode1D component if the database support is required.

3. Create the OnBeforePrint event function for the TppReport component.

   In the event function, change the properties of the TBarcode1D component such as Barcode, Module, and Ratio, and adjust the bitmap size of the TppImage control in order to accommodate entire barcode symbol, then use the DrawTo method of the TBarcode1D component to draw the barcode symbol to the TppImage control.

   For example:

```
var
  BarcodeWidth, BarcodeHeight, SymbolWidth, SymbolHeight: Integer;
begin
  ......
  Barcode1D_Code391.Barcode := '1235678';
  Barcode1D_Code391.Module := 2;
  ......
  with ppReport1Image1.Picture.Bitmap do
  begin
    Barcode1D_Code391.DrawToSize(BarcodeWidth, BarcodeHeight, SymbolWidth, SymbolHeight, Canvas);
    Width := BarcodeWidth;
    Height := 100;
    Barcode1D_Code391.DrawTo(Canvas, 0, 0);
  end;
```

3.6 How to use the barcode components with ACE Reporte

Usage:

1. Edit your report, put a TsctImageLabel control to your report in order to present the barcode symbol.
2. Put a TBarcode1D barcode component, such as the TBarcode1D_Code39, TBarcode1D_EAN13, and TBarcode1D_Code128 to your form that the TsctReport component is in it.

   Also, put a TDBBarcode1D component to the form and link the TBarcode1D component to the TDBBarcode1D component if the database support is required.

3. Create the OnBeforePrint event function for the TsctImageLabel control.

   In the event function, change the properties of the TBarcode1D component such as Barcode, Module, and Ratio, and adjust the bitmap size of the TsctImageLabel control in order to accommodate entire barcode symbol, then use the DrawTo method of the TBarcode1D component to draw the barcode symbol to the TsctImageLabel control.

   For example:

```
var
  BarcodeWidth, BarcodeHeight, SymbolWidth, SymbolHeight: Integer;
begin
  ......
  Barcode1D_Code391.Barcode := '1235678';
  Barcode1D_Code391.Module := 2;
  ......
  with SctImageLabel1.Picture.Bitmap do
  begin
    Barcode1D_Code391.DrawToSize(BarcodeWidth, BarcodeHeight, SymbolWidth, SymbolHeight, Canvas);
    Width := BarcodeWidth;
    Height := 100;
    Barcode1D_Code391.DrawTo(Canvas, 0, 0);
  end;
```

3.7 How to use the barcode components with Rave Reports

Usage:

1. Use the Rave Reports Visual Designer to edit your report, put a bitmap component to your report in order to present the barcode symbol.
2. Put a TBarcode1D barcode component, such as the TBarcode1D_Code39, TBarcode1D_EAN13, and TBarcode1D_Code128 to your form that the TRvProject component is in it.

   Also, put a TDBBarcode1D component to the form and link the TBarcode1D component to the TDBBarcode1D component if the database support is required.

3. Insert code to call the Open method of the TRvProject component before print or preview the report, and call the Close method after print or preview the report.

   For example:

```
RvProject1.Open;
RvProject1.Execute;
RvProject1.Close;
```

4. Add RvCsStd, RvProj, and RvClass units to the uses list.
5. Create the OnAfterOpen event function for the TRvProject component.

   In the event function, change the properties of the TBarcode1D component such as Barcode, Module, and Ratio. Then create a temporary TBitmap object, adjust its size in order to accommodate entire barcode symbol, and use the DrawTo method of the TBarcode1D component to draw the barcode symbol to the temporary TBitmap object. At last, adjust the size of the bitmap component in your report, and assign the temporary bitmap to it.

   For example:

```
var
  RvReport: TRaveReport;
  RvPage : TRavePage;
  RvBitmap: TRaveBitmap;
  TmpBitmap: TBitmap;
  BarcodeWidth, BarcodeHeight, SymbolWidth, SymbolHeight: Integer;
  Scale: Integer;
begin
  ......
  Barcode1D_Code391.Barcode := '1235678';
  Barcode1D_Code391.Module := 2;
  ......
  with RvProject1.ProjMan do
  begin
    RvReport := FindRaveComponent('Report1', nil) as TRaveReport;
    RvPage := FindRaveComponent('Page1', RvReport) as TRavePage;
    RvBitmap := FindRaveComponent('Bitmap1', RvPage) as TRaveBitmap;
  end;
  TmpBitmap := TBitmap.Create;
  try
    Barcode1D_Code391.DrawToSize(BarcodeWidth, BarcodeHeight, SymbolWidth, SymbolHeight, TmpBitmap.Canvas);
    TmpBitmap.Width := BarcodeWidth;
    TmpBitmap.Height := BarcodeHeight;
    Scale := 72;
    RvBitmap.Width := RvReport.XI2U(BarcodeWidth / Scale);
    RvBitmap.Height := RvReport.YI2U(BarcodeHeight / Scale);
    RvBitmap.MatchSide := msBoth;
    Barcode1D_Code391.DrawTo(TmpBitmap.Canvas, 0, 0);
    RvBitmap.Image.Assign(TmpBitmap);
  finally
    TmpBitmap.Free;
  end;
  ......
end;
```

3.8 How to print a barcode symbol to paper

Please use the **Print** method of the barcode component to print the barcode symbol to paper. The TImage control isn't required.

For example:

```
var
  ...
  TextDefine: TBarcodeTextDefine;
begin
  ...
  Printer.BeginDoc;
  ... { Print other content }
  Font1 := TFont.Create;
  try
    Font1.Name := 'Comic Sans MS';
    Font1.Size := 9;
    TextDefine.DisplayText := dtBarcode;
    TextDefine.TextPosition := tpBottomOut;
    TextDefine.TextAlignment := taJustify;
    TextDefine.TextFont := Font1;
    TextDefine.ExtraFontSize := 9;
    Barcode1D_Code391.Print(20, 20, '1234567890', True, clBlack, clWhite, TextDefine, 2, 0.3);
  finally
    Font1.Free;
  end;
  ... { Print other content }
  Printer.EndDoc;
  ...
```

```
  end;
```

or

```
  var
    ...
    TextDefine: TBarcodeTextDefine;
  begin
    ...
    Printer.BeginDoc;
    ... { Print other content }
    with Barcode1D_Code391 do
    begin
      Ratio := 2;
      AutoCheckDigit := True;
      BarColor := clBlack;
      SpaceColor := clWhite;
      DisplayText := dtBarcode;
      TextPosition := tpBottomOut;
      TextAlignment := taJustify;
      TextFont.Name := 'Comic Sans MS';
      TextFont.Size := 9;
      Barcode := '1234567890';
      Print(20, 20, 0.3);
    end;
    ... { Print other content }
    Printer.EndDoc;
    ...
  end;
```

3.9 How to save a barcode symbol to picture file

Please use the **SaveToFile** method of the TImage control that is linked to a TBarcode1D barcode component to save the barcode symbol as a picture file.

For example:

```
  Image1.Picture.Bitmap.SaveToFile('C:\barcode.bmp');
```

3.10 How to encode the UNICODE text in a Code128/EAN128 symbol

By default, the text will be encoded in ANSI ecnoding sheme, you can use other encoding scheme, such as the UTF-8, UTF-16LE, UTF-16BE, etc.

Note, the feature isn't supported by almost barcode scanners.

- For Delphi/C++ Builder 2007 or early:

  Method 1, please convert the text to your encoding scheme, then assign it to the Barcode property of the barcode component. The BOM may be placed depending on your application. And then create the OnDecodeText event function in order to decode the barcode text from your encoding string and output it into the symbol.

  For the TBarcode1D_Code128 component, please change the AutoCheckDigit property to true. For the TBarcode1D_EAN128 component, please change the AutoCheckDigit property to false.

  For example:

```
  var BarcodeText: string;
  ....
  BarcodeText := '....';
  // The text is encoded in UTF-8 format, and the BOM is placed.
  Barcode1D_Code1281.AutoCheckDigit := True;
  Barcode1D_Code1281.Barcode := #$EF#$BB#$BF + AnsiToUTF8(BarcodeText);
  ....
  procedure TForm1.Barcode1D_Code1281DecodeText(Sender: TObject; var BarcodeText: string; Data: AnsiString);
  begin
    // Remove the BOM before decoding.
    BarcodeText := UTF8ToAnsi(Copy(Data, 4, Length(Data) - 3);
  end;
```

  Method 2, Please create the OnEncode event function for the barcode component. In the event function, you can encode the UNICODE text in your encoding scheme. The BOM may be placed depending on your application.

For the TBarcode1D_Code128 component, please change the AutoCheckDigit property to true. For the TBarcode1D_EAN128 component, please change the AutoCheckDigit property to false.

For example:

```
var BarcodeText: string;
....
BarcodeText := '....';
Barcode1D_Code1281.AutoCheckDigit := True;
Barcode1D_Code1281.Barcode := BarcodeText;
....
procedure TForm1.Barcode1D_Code1281Encode(Sender: TObject; var Data: AnsiString; Barcode: string);
begin
  // The text is encoded in UTF-8 format, and the BOM is placed.
  Data := #$EF#$BB#$BF + AnsiToUTF8(Barcode);
end;
```

- For Delphi/C++ Builder 2009 or later:

  Method 1, please convert the text to your encoding scheme, then assign it to the Data property of the barcode component. The BOM may be placed depending on yourapplication. And then create the OnDecodeText event function in order to decode the barcode text from your encoding string and output it into the symbol.

  For the TBarcode1D_Code128 component, please change the AutoCheckDigit property to true. For the TBarcode1D_EAN128 component, please change the AutoCheckDigit property to false.

  For example:

```
// The text is encoded in UTF-8 format, and the BOM is placed.
var BarcodeText: string;
....
BarcodeText := '....';
Barcode1D_Code1281.AutoCheckDigit := True;
Barcode1D_Code1281.Data := #$EF#$BB#$BF + UTF8Encode(BarcodeText);
....
procedure TForm1.Barcode1D_Code1281DecodeText(Sender: TObject; var BarcodeText: string; Data: AnsiString);
begin
  // Remove the BOM before decoding.
  BarcodeText := UTF8Decode(Copy(Data, 4, Length(Data) - 3);
end;
```

  Method 2, Please create the OnEncode event function for the barcode component. In the event function, you can encode the UNICODEtext in your encoding scheme. The BOM may be placed depending on your application.

  For the TBarcode1D_Code128 component, please change the AutoCheckDigit property to true. For the TBarcode1D_EAN128 component, please change the AutoCheckDigit property to false.

  For example:

```
var BarcodeText: string;
....
BarcodeText := '....';
Barcode1D_Code1281.AutoCheckDigit := True;
Barcode1D_Code1281.Barcode := BarcodeText;
....
procedure TForm1.Barcode1D_Code1281Encode(Sender: TObject; var Data: AnsiString; Barcode: string);
begin
  // The text is encoded in UTF-8 format, and the BOM is placed.
  Data := #$EF#$BB#$BF + UTF8Encode(Barcode);
end;
```

Chapter 4. Reference

4.1 TBarcode1D

It is the base class of all barcode components, and defined in the pBarcode1D unit.

Properties:

- Barcode (Protected)
- Data (*)
- Ratio (Protected)
- AutoCheckDigit (Protected)
- TextHSpacing (Protected)
- ExtraFontSize (Protected)
- FullEncoded (Public)
- Image
- Module
- Height
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextVSpacing
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.1 TBarcode1D_AP4SC

The component is used to create the Australia Post 4-State Customer Barcodes symbol. It's defined in the pAP4SC unit.

The barcode is also known as the 4-State barcode. It is used by Australia Post for Postal code and automatic mail sorting.

Technical Details:

The Australia Post uses seven different formats of PostBar codes. They are specified in the following:

- **Standard Customer Barcode:** <FCC><DPID>
  - **FCC:** Format Control Code Field

    Character "11"
  - **DPID:** Sorting Code Field (Delivery Point Identifier)

    Length: 8 characters; Characters set: "0"-"9"

  Example: 1139987520
- **Customer Barcode 2:** <FCC><DPID><CI>
  - **FCC:** Format Control Code Field

Character "59"

- **DPID:** Sorting Code Field (Delivery Point Identifier)

  Length: 8 characters; Characters set: "0"-"9"

- **CI:** Customer Information Field
  - Property NumCustomerInfo is set to true:

    Length: 0-8 characters; Characters set: "0"-"9"
  - Property NumCustomerInfo is set to false:

    Length: 0-5 characters; Characters set: "0"-"9", "A"-"Z", "a"-"z", "#" and " " (space)

Example: 5932211324A124B

- **Customer Barcode 3:** <FCC><DPID><CI>
  - **FCC:** Format Control Code Field

    Character "62"
  - **DPID:** Sorting Code Field (Delivery Point Identifier)

    Length: 8 characters; Characters set: "0"-"9"
  - **CI:** Customer Information Field
    - Property NumCustomerInfo is set to true:

      Length: 0-15 characters; Characters set: "0"-"9"
    - Property NumCustomerInfo is set to false:

      Length: 0-10 characters; Characters set: "0"-"9", "A"-"Z", "a"-"z", "#" and " " (space)

Example: 6282224535CAM555439

- **Reply Paid Barcode:** <FCC><DPID>
  - **FCC:** Format Control Code Field

    Character "45"
  - **DPID:** Sorting Code Field (Delivery Point Identifier)

    Length: 8 characters; Characters set: "0"-"9"

Example: 4567671415

- **Routing Barcode:** <FCC><DPID>
  - **FCC:** Format Control Code Field

    Character "87"
  - **DPID:** Sorting Code Field (Delivery Point Identifier)

    Length: 8 characters; Characters set: "0"-"9"

Example: 8756439111

- **Redirection Barcode:** <FCC><DPID>
  - **FCC:** Format Control Code Field

    Character "92"
  - **DPID:** Sorting Code Field (Delivery Point Identifier)

    Length: 8 characters; Characters set: "0"-"9"

Example: 9235797531

- **Currently Reserved:** <FCC><DPID>
  - **FCC:** Format Control Code Field

    Character "44"
  - **DPID:** Sorting Code Field (Delivery Point Identifier)

    Length: 8 characters; Characters set: "0"-"9"
  - **CI:** Customer Information Field
    - Property NumCustomerInfo is set to true:

      Length: 0-15 characters; Characters set: "0"-"9"
    - Property NumCustomerInfo is set to false:

      Length: 0-10 characters; Characters set: "0"-"9", "A"-"Z", "a"-"z", "#" and " " (space)

Example: 4448487312ABCDEFGHIJ

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- NumCustomerInfo
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- SplitText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.2 TBarcode1D_BC309

The component is used to create the BC309 barcode symbol. It's defined in the pBC309 unit.

The BC309 barcode symbology is a variant of the BC412 barcode. It was developed by Semiconductor Equipment and Materials International (SEMI). It's a single-width numeric-only barcode symbology.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Variable
- **Code type:** Clocked 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextHSpacing
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.3 TBarcode1D_BC412

The component is used to create the BC412 barcode symbol. It's defined in the pBC412 unit.

The BC412 barcode symbology was developed by Semiconductor Equipment and Materials International (SEMI) in 1993. The relatively new BC412 barcode was developed by Computer Identics and IBM in 1988. Used to mark the serial numbers on semiconductor wafers. It's a single-width barcode.

Technical Details:

- **Characters set:** 0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ
- **Length:** Variable
- **Code type:** Clocked 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextHSpacing
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.4 TBarcode1D_Channel

The component is used to create the Channel Code barcode symbol. It's defined in the pChannel unit.

The Channel Code barcode symbology is a linear, continuous, self-checking, bidirectional symbology that encodes between 2 and 7 digits in the least symbol length possible.

The channel barcode symbolpgy has 6 channels from 3 to 8, it is determined to be one more than the number of digits given in the barcode text. Corresponding to these different channels, the barcode symbolpgy can hold numbers from any of the following ranges:

- **Channel 3**: 0-26
- **Channel 4**: 0-292
- **Channel 5**: 0-3493
- **Channel 6**: 0-44072
- **Channel 7**: 0-576688
- **Channel 8**: 0-7742862

Technical Details:

- **Characters set:** "0"-"9"
- **Length:** Variable
- **Code type:** Linear continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- ShortFinder
- Channel
- CurrentChannel
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.5 TBarcode1D_Clocked35

The component is used to create the Clocked 35 barcode symbol. It's defined in the pClocked35 unit.

The Clocked 35 barcode symbology is known as China Postal Code, Korean Postal Authority. It is a clocked code consisting of a 6-digits ZIP code plus a check digit. In Korean, the ZIP code may be provided with a dash between the first three and last three digits, and the dash is not encoded.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 6 characters, othwise 7 characters (the check digit can be specified by you)
- **Code type:** Clocked 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextHSpacing
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.6 TBarcode1D_Codabar

The component is used to create the Codabar barcode symbol. It's defined in the pCodabar unit.

The Codabar barcode symbology is also known as ABC Codabar, CodaBar, USD-4, NW-7, Code 2 of 7, Monarch, Code-27, Ames code, Rationalized Codabar, 2 of 7 Code, ANSI/AIM Codabar, Uniform Symbology Specification Codabar, USS Codabar, etc. It was developed in 1972 by Pitney Bowes, Inc. is a discrete, variable self-checking symbology that may encode 16 different characters. There are four different start and stop signs defined. They are only valid at the beginning and the end of the code. They can be used to transport additional information. This barcode symbology is used by U.S. blood banks, photo labs, and on FedEx airbills.

Technical Details:

- **Characters set:** 0123456789-$:/.+
- **Length:** Variable
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- StartCode
- StopCode
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.7 TBarcode1D_Code11

The component is used to create the Code 11 barcode symbol. It's defined in the pCode11 unit.

The Code 11 barcode symbology is also known as Code11, USD-8, USD8, etc. It was developed as a high-density variable-length numeric-only symbology. The symbology is discrete and is able to encode the numbers 0 through 9, and the dash symbol (-). It is used primarily in labeling telecommunications equipment. Code 11 is not terribly secure in that printing imperfections can quite easily convert one character into another valid character. Data integrity is obtained by using one, or sometimes two, check characters.

Technical Details:

- **Characters set:** 0123456789-
- **Length:** Variable
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- NumberCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.8 TBarcode1D_Code128

The component is used to create the Code128 barcode symbol. It's defined in the pCode128 unit.

The Code 128 barcode symbology is also known as ANSI/AIM 128, ANSI/AIM Code 128, USS Code 128, Uniform Symbology Specification Code 128, Coc 128 Code Set A, Code 128 Code Set B, Code 128 Code Set C, Code 128A, Code 128B, Code 128C, etc. It is is a very high-density barcode symbology. I is used for alphanumeric or numeric-only barcodes.

It can encode all 128 characters of ASCII character sets. This is done by switching between all 3 character subsets of Code 128:

- **SubSet A:** Includes characters with ASCII values from 00 to 95 (i.e. all of the standard upper case alphanumeric characters together with the control characters inclusive), and function characters.
- **SubSet B:** Includes characters with ASCII values from 32 to 127 (i.e. all of the standard upper case alphanumeric characters together with the lower case alphabetic characters inclusive), and function characters.
- **SubSet C:** includes the set of 100 digit pairs from 00 to 99 inclusive, as well as seven special characters. This allows numeric data to be encoded, two data digits per symbol character, at effectively twice the density of standard data.

  Note, only an even number of digits can be encoded if using the character subset C.

If the EncodeMode property is set to **cemAuto**, the character subset will be switched automatically in a Code 128 symbol in order to minimize the symbol size. If it is set to **cemManual**, you need to switch the character subset manually by using following function symbols:

- **CODE A**: Switch to character subset A. Please use the escape sequence "**\a**" to insert the symbol.
- **CODE B**: Switch to character subset B. Please use the escape sequence "**\b**" to insert the symbol.
- **CODE C**: Switch to character subset C. Please use the escape sequence "**\c**" to insert the symbol.
- **SHIFT**: Change the character subset from A to B or B to A for the single character following the "**SHIFT**" escape sequence. Please use the escape sequence "**\s**" to place the symbol to barcode text.

Also, you can manually switch the character subset by using these function symbols even if the EncodeMode property is set to **cemAuto**.

Characters with ASCII values 128 to 255 in accordance with ISO 8859-1:1998 may also be encoded. This is done by using the **FNC 4**" symbol together with character subsets A, B and C. If the EncodeMode property is set to **cemAuto**, the "**FNC 4**' will be inserted automatically depending on the barcode text in order to minimize the symbol size. If it is set to **cemManual**, you need to insert the "**FNC 4**" symbol manually by using "**\4**".

If a single "**FNC 4**" character is used, indicates the following data character in the symbol is a extended ASCII character. A "**SHIFT**" character may follow the "**FNC 4**" character if it is necessary to change character subset for the following data character. Subsequent data characters revert to the standard ASCII character set. If two consecutive "**FNC4**" characters are used, all following data characters are extended ASCII characters until two further consecutive "**FNC4**" characters are encountered or the end of the symbol is reached. If during this sequence of extended encodation a single "**FNC4**" character is encountered it is used to revert to standard ASCII encodation for the next data character only. "**SHIFT**" and character subset characters shall have their normal effect during such a sequence.

Also, the "**FNC 1**", "**FNC 2**", and "**FNC 3**" symbols can be used for special purposes. You can use the escape sequences "**\1**", "**\2**", and "**\3**" to place them to the barcode text.

Note, if you want to place the "****" character to barcode text, please use the "****" escape sequence.

See also the InitialSubSet and the EncodeMode properties.

Technical Details:

- **Characters set:** All 128 ASCII characters (ASCII 0 - ASCII 127), characters with ASCII values 128 to 255 may also be encoded
- **Length:** Variable
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- EncodeMode
- InitialSubSet
- CurrentSubSet
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnEncode (*)
- OnDecodeText (*)
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnEncode, OnDecodeText, OnInvalidDataLength, and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.9 TBarcode1D_Code25Datalogic

The component is used to create the Code 25 Datalogic barcode symbol. It's defined in the pCode25Dat unit.

The Code 25 Datalogic barcode symbology is also known as China Postal code, etc. It is a higher-density variable-length numeric-only barcode symbology based upon the Code 25 Matrix barcode symbology. It is used primarily for China postal.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Variable
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.10 TBarcode1D_Code25Industrial

The component is used to create the Code 25 Industrial barcode symbol. It's defined in the pCode25Ind unit.

The Code 25 Industrial barcode symbology is also known as Industrial 2 of 5, 2 of 5 Industrial, 2/5 Industrial, 2 of 5 Standard, Standard 2 of 5, 2/5 Standard, Code 2/5, 2 of 5, C 2 of 5, Code25, Discrete 2 of 5, etc. It is a low-density variable-length numeric-only symbology that has been with us since the 1960s. It has been used in the photofinishing and warehouse sorting industries, as well as sequentially numbering airline tickets.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Variable
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.11 TBarcode1D_Code25Interleaved

The component is used to create the Code 25 Interleaved, the USPS Sack Label (USPS 25 Sack Label), or the USPS Sack Label (USPS 25 Sack Label) barco symbol. It's defined in the pCode25Int unit.

The Code 25 Interleaved barcode symbology is also known as Interleaved 2 of 5, ANSI/AIM ITF 25, ANSI/AIM I-2/5, Uniform Symbology Specification ITF, USS I 2/5, ITF, I-2/5, 2 of 5 Interleaved, 2/5 Interleaved, etc. It is a higher-density variable-length numeric-only barcode symbology based upon the Industrial 2 of 5 symbology. It encodes digit pairs in an interleaved manner. The odd position digits are encoded in the bars and the even position digits are encoded in the spaces. Because of this, It must consist of an even number of digits. It is suitable for encoding general purpose all-numeric data and is used primarily in the distribution and warehouse industry.

The USPS Sack Label symbology (USPS 25 Sack Label) is in fact Code 2 of 5 Interleaved symbology with exactly 8 digits encoded: 5-digit ZIP Code (the sack destination) and a 3-dig content identifier number(CIN).

The USPS Tray Label symbology (USPS 25 Tray Label) is in fact Code 2 of 5 Interleaved symbology with exactly 10 digits encoded: 5-digit ZIP Code (the tray destination) and a 3-digi content identifier number(CIN), and a 2-digit USPS processing code.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Variable
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- Padding
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.12 TBarcode1D_Coed25Invert

The component is used to create the Code 25 Invert barcode symbol. It's defined in the pCode25Inv unit.

The Code 25 Invert barcode symbology is known as Invert, 25 Invert, 2of 5 Invert, 2/5 Invert, etc. It is a low-density variable-length numeric-only symbology. And it is a variation of Code 25 Industrial barcode symbology.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Variable
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.13 TBarcode1D_Code25Matrix

The component is used to create the Code 25 Matrix barcode symbol. It's defined in the pCode25Mat unit.

The Code 25 Matrix barcode symbology is also known as Matrix 2 of 5, 2 of 5 Matrix, 2/5 Matrix, etc. It is a higher-density variable-length numeric-only barcode symbology based upon the Code 25 Industrial barcode symbology. It is used primarily for warehouse sorting, photo finishing, and airline ticket marking.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Variable
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.14 TBarcode1D_Code32

The component is used to create the Code 32 barcode symbol. It's defined in the pCode32 unit.

The Code 32 barcode symbology is also known as Italian Pharmacode, IMH, Codice 32 Pharmacode, Codice Farmaceutico Italiano, Radix 32 Barcode, etc. It is mainly used to encode pharmaceutical products in Italy. It has the following structure:

1. An 'A' character (ASCII 65) which is not really encoded, It does not need to be entered in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property.
2. 8 digits for Pharmacode (It generally begins/is prefixed with 0 - zero).
3. 1 digit for Checksum, If the AutoCheckDigit property is set to true, it doesn't need to be entered in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, and it will be automatically calculated.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 8 characters, othwise 9 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- ShowGuards
- InterGap
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.15 TBarcode1D_Code39

The component is used to create the Code 39 (LOGMARS) or the ESN barcode symbol. It's defined in the pCode39 unit.

The Code 39 barcode symbology is also known as ANSI/AIM Code 39, Uniform Symbology Specification Code 39, USS Code 39, USS 39 Code 3/9, Code 3 of 9, USD-3, LOGMARS, Alpha39, etc.

Code 39 barcode symbology, the first alpha-numeric symbology to be developed, is still widely used-especially in non-retail environments. It is suitable for encoding general purpose alphanumeric data. Code 39 is a discrete, variable-length symbology. It is self-checking in that a single print defect cannot transpose one character into another valid character. It is the standard barcode used by the United States Department of Defense, and is also used by the Health Industry Bar Code Council (HIBCC).

Also, you can use the component to create the Numly Number barcode symbol. A Numly Number is an ESN or Electronic Serial Number for all things digital. It is a unique identifier that allows an author or publisher to assign to content and track licensing of each id assignment. Numly Numbers are useful if you wish to identify each electronic distributed copy of any form of electronic media. Media types could include: Blogs, Emails, MP3s, Videos, PDFs, eBooks, Software, Websites, etc. Numly Numbers can also act a third-party content submission time stamps to aid in copyright proving instances and emails. The Numly Number consists of a 19 digit number generated by an algorithm maintained by Numly.com.

Technical Details:

- **Characters set:** "0"-"9", "A"-"Z", "-", ".", "$", "/", "+", "%", and " " (space)
- **Length:** Variable
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- InterGap
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- ShowGuards
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.16 TBarcode1D_Code39Ext

The component is used to create the Code 39 Extended barcode symbol. It's defined in the pCode39Ext unit.

The Code 39 Extended barcode symbology is also known as Code 39 Full ASCII, etc. It is an extended version of Code 39 that supports all 128 ASCII characters. So with Code 39 Extended you can also code the 26 lower letters (a-z) and other special characters. The additional characters (e.g. lower case letters) are created using the existing characters of Code 39 by combining two characters each.

Technical Details:

- **Characters set:** All 128 ASCII characters (ASCII 0 - ASCII 127)
- **Length:** Variable
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- InterGap
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- ShowGuards
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.17 TBarcode1D_Code93

The component is used to create the Code 93 barcode symbol. It's defined in the pCode93 unit.

The Code 93 barcode symbology is also known as ANSI/AIM Code 93, ANSI/AIM Code 93, Uniform Symbology Specification Code 93, USS Code 9 USS 93, Code 9/3, USS-93, USD-3, etc. It is a continuous, higher-density variable-length barcode symbology. It was designed to complement and improve upon Code 39. It offers higher information density for alphanumeric data than Code 39. Code 93 also incorporates two check digits as an added measure of security. Although Code 93 is considered more robust than Code 39, it has never achieved the same popularity as Code 39.

Technical Details:

- **Characters set:** "0"-"9", "A"-"Z", "-", ".", "$", "/", "+", "%", and " " (space)
- **Length:** Variable
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.18 TBarcode1D_Code93Ext

The component is used to create the Code 93 Extended barcode symbol. It's defined in the pCode93Ext unit.

The Code 93 Extended barcode symbology is also known as Code 39 Full ASCII, etc. It is an extended version of Code 93 that supports all 128 ASCII characters. The characters represented by Code 93 are represented in Code 93 Extended as single bar code characters, but all other characters are represented by a control character plus another character.

Technical Details:

- **Characters set:** All 128 ASCII characters (ASCII 0 - ASCII 127)
- **Length:** Variable
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.19 TBarcode1D_Coop25

The component is used to create the COOP 25 barcode symbol. It's defined in the pCoop25 unit.

The COOP 25 barcode symbology is also known as Coop 2 of 5, Coop 2/5, Coop25, NEC 25, NEC 2 of 5, NEC 2/5, NEC25, etc. It is a higher-density variable length numeric-only barcode symbology. It was adopted for management of Physical Distribution "Process 8000".

Technical Details:

- **Characters set:** 0123456789
- **Length:** Variable (COOP is fixed to 4-digit)
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.20 TBarcode1D_CPCBin

The component is used to create the CPC Binary barcode symbol. It's defined in the pCPCBin unit.

The CPC Binary barcode symbology is Canada Post's proprietary symbology used in its automated mail sortation operations. It encodes the destination postal code. This barcode is used on regular-size pieces of mail, especially mail sent using Canada Post's Lettermail service. This barcode is printed on the lower-right-hand corner of each faced envelope, using a unique ultraviolet-fluorescent ink.

Technical Details:

- **Format:** ANANAN
- **Characters set:** A: ABCEGHJKLMNPRSTVWXYZ(The DFIOQU are invalid characters); N: 0123456789
- **Length:** Fixed (6 characters)
- **Code type:** Clocked 1D Continuous
- **Example:** L3B4T9

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextHSpacing
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.21 TBarcode1D_EAN128

The component is used to create the EAN-128 barcode symbol. It's defined in the pEAN128 unit.

The EAN-128 barcode symbology is also known as UCC/EAN-128, UCC-128, USS-128, GS1-128, UCC.EAN-128, GTIN-12 UCC-12, EAN/UCC-13, EAN/UCC-14, etc. It is a subset of the more general Code 128 symbology. It was developed to provide a worldwide format and standard for exchanging common data between companies. An EAN-128 barcode is a Code 128 barcode and it is constructed like this:

1. **FNC 1:** Function Character 1 in Code 128 symbology, It does not need to be entered in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, and it will be automatically inserted. And it is not represented in the human readable text.
2. **AI:** Application identifier, It is a 2, 3, or 4-digits number that identifies the type and format of the data which follows. By convention, the AI is enclosed in parentheses when printed below the barcode (the parentheses are only used for human readable text, and are not encoded in the barcode).
3. **Data:** The data field, Its meaning and format are identified by the AI. For some AI fields, an extra check digit is required, you can set the AutoCheckDigit property to true to automatically calculate the check digit, and use the CheckStart and the CheckLength properties to decide which characters will be used to calculate the check digit.
4. **Code 128 check digit:** Check digit in Code 128 symbology. It does not need to be entered in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, and it will be automatically calculated even if the AutoCheckDigit property is set to false.

A single barcode may contain more than one type of information. The beginning of each new piece of information is marked by an AI. An FNC1 is required after each variable-length field (do not use FNC1 after the last data field, and do not use FNC1 if the maximal field length is used). For example, An EAN-128 barcode contains two pieces and the first pieces is variable-length field. It is constructed like this:

1. **FNC 1:** Function Character 1 in Code 128 symbology, It does not need to be entered in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, and it will be automatically inserted. And it is not represented in the human readable text.
2. **AI:** First application identifier.
3. **Data:** First data field. Its meaning and format are identified by first AI.
4. **FNC 1:** Function Character 1 in Code 128 symbology, It need to be entered in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, and it is not represented in the human readable text.
5. **AI:** Second application identifier.
6. **Data:** Second data field. Its meaning and format are identified by second AI.
7. **Code 128 check digit:** Check digit in Code 128 symbology. It does not need to be entered in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, and it will be automatically calculated even if the AutpCheckDigit property is set to false.

Note, the CheckStart and the CheckLength properties only work for last data field. For the CheckStart property, the index of first AI's first character is 1.

See also the TBarcode1D_Code128 component.

The EAN128 barcode symbol can be used together with a 4-column CC-A, a 4-column CC-B or a CC-C 2D symbol to create the EAN.UCC composite symbol. If you use the componer together with the **CC-A**, **CC-B** or **CC-C** 2D barcode component that's in our **2D Barcode VCL Components** package, it can be used as the **Liner** property's value of the **CC-A**, **CC-B** or **CC-C** 2D barcode component to create the EAN.UCC composite symbols. In such case, only the Barcode, Data (only for Delphi/C++ Builder 2009 or later), AutoCheckDigit, CheckStart, CheckLength, InitialSubSet, Height, DisplayText, TextPosition, TextAlignment, TextFont, TextVSpacing, and the TextHSpacing properties are useful, the value of other properties will be ignored and they will be set automatically depending on the settings of the 2D barcode component (the Link2D property will be set automatically depending on which 2D component is used). In addition, the Height property should be set to a value larger than zero, and the Image should not be normally set. If you use it together with other 2D components package, the Link2D property should be set by yourself.

Technical Details:

- **Characters set:** All 128 ASCII characters (ASCII 0 - ASCII 127), characters with ASCII values 128 to 255 may also be encoded
- **Length:** Variable
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- CheckStart
- CheckLength
- EncodeMode
- InitialSubSet
- CurrentSubSet
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Link2D
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnEncode (*)
- OnDecodeText (*)
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnEncode, OnDecodeText, OnInvalidDataLength, and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.22 TBarcode1D_EAN13

The component is used to create the EAN-13, ISBN, ISMN, ISSN or the JAN-13 barcode symbol. It's defined in the pEAN13 unit.

The EAN-13 barcode symbology is also known as European Article Number 13, EAN13, UPC-13, GTIN-13, GS1-13, EAN/UCC-13, etc. It is used world-wide for marking retail goods. The symbolology encodes 13 digits. The value to encode by EAN-13 has the following structure:

1. 2 or 3 digits for Number System or Country Code.
2. 5 or 4 digits for Manufacturer (Company) Code or prefix.
3. 5 digits for Product Code.
4. 1 digit for checksum.

An EAN-2 or an EAN-5 barcode may be added for a total of 14 or 17 data digits.

The ISBN symbology is also known as International Standard Book Number, Bookland EAN, ISBN-13, ISBN-10, etc. It is created using the EAN-13 symbology with a special prefix, f example the prefix 978. So the ISBN is a special form of the EAN-13 code. An EAN-5 supplemental barcode symbol is usually used for the Retail Suggested Price.

The ISMN symbology is also known as International Standard Music Number, ISMN-13, ISMN-10, ISO 10957, etc. It is a unique number for the identification of all printed musi publications from all over the world. It is created using the EAN-13 symbology with a special prefix, for example the prefix 979. So the ISMN is a special form of the EAN-13 code. An EAN-5 supplemental barcode symbol is usually used for the Retail Suggested Price.

The ISSN symbology is also known as International Standard Serial Number, ISSN-13, ISSN-10, etc. It is a unique number which identifies periodical publications as such, including electronic serials. It is created using the EAN-13 symbology with a special prefix, for example the prefix 977. So the ISSN is a special form of the EAN-13 code. An EAN-2 supplemental barcode symbol is usually used to indicate an issue number.

The JAN13 symbology is mostly the same as EAN-13 symbology, but used in Japan. JAN stands for Japanese Numbering Authority. First two digits of JAN-13 symbology are always "49".

The EAN-13 barcode symbol can be used together with a 4-column CC-A or a 4-column CC-B 2D symbol to create the EAN.UCC composite symbol. If you use the **2D Barcode VCL Components** package, the component can be used as the value of the **Liner** property of the **CC-A** or **CC-B** 2D barcode component to create the EAN.UCC composite symbols. In such case, only the Barcode, Data (only for Delphi/C++ Builder 2009 or later),AutoCheckDigit, Height, DisplayText, TextPosition, TextAlignment, TextFont, TextVSpacing, TextHSpacing, ExtraFontSize, LeftQuietZoneSpacing, and the RightQuietZoneSpacing properties are useful, the value of other properties will be ignored and they will be set automatically depending on the settings of the 2D barcode component. In addition, the Height property should be set to a value larger than zero, and the Image should not be normally set.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 12 characters, othwise 13 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- ShowQuietZoneMark
- LeftQuietZoneSpacing
- RightQuietZoneSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.23 TBarcode1D_EAN2

The component is used to create the EAN/UPC Add On 2 barcode symbol. It's defined in the pEAN2 unit.

The EAN/UPC Add On 2 barcode symbology is also known as EAN Supplement 2/Two-digit Add-On, EAN+2, UPC Supplement 2/Two-digit Add-On, UPC+2, etc. It is designed to encode information supplementary to that in the main barcode symbol on periodicals, hardback, and paperback books. It may be used specific applications to accompany an EAN-8, EAN-13, UPC-A, or UPC-E (including UPC-E0 and UPC-E1) barcodes. It is often used on magazines and periodicals to indicate an issue number. In general, it is positioned following the right Quiet Zone of the main barcode symbol.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. 2 digits
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- ShowQuietZoneMark
- LeftQuietZoneSpacing
- RightQuietZoneSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.24 TBarcode1D_EAN5

The component is used to create the EAN/UPC Add On 5 barcode symbol. It's defined in the pEAN5 unit.

The EAN/UPC Add On 2 barcode symbology is also known as EAN Supplement 5/Five-digit Add-On, EAN+5, UPC Supplement 5/Five-digit Add-On, UPC+5, etc. It designed to encode information supplementary to that in the main barcode symbol on periodicals, hardback, and paperback books. It may be used specific applications to accompany an EAN-8, EAN-13, UPC-A, or UPC-E (including UPC-E0 and UPC-E1) barcodes. In general, it is positioned following the right Quiet Zone of the main barcode symbol. And It is often used for the price of books together with the ISBN code.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. 5 digits
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- ShowQuietZoneMark
- LeftQuietZoneSpacing
- RightQuietZoneSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.25 TBarcode1D_EAN8

The component is used to create the EAN-8 barcode symbol. It's defined in the pEAN8 unit.

The EAN-8 barcode symbology is also known as European Article Number 8, EAN8, UPC-8, GTIN-8, GS1-8, EAN/UCC-8, etc. It is a shortened version of the EAN-13 code that is intended to be used on packaging which would be otherwise too small to use one of the other versions. EAN-8 barcodes may be used to encode GTIN-8s which are another set of product identifiers from the GS1 System. The value to encode by EAN-8 has the following structure:

1. 2 or 3 digits for GS1 prefix.
2. 5 or 4 digits for Item reference element depending on the length of the GS1 prefix.
3. 1 digit for checksum.

An EAN-2 or an EAN-5 barcode may be added for a total of 10 or 13 data digits.

The EAN-8 barcode symbol can be used together with a 3-column CC-A or a 3-column CC-B 2D symbol to create the EAN.UCC composite symbol. If you use the **2D Barcode VCL Components** package, the component can be used as the value of the **Liner** property of the **CC-A** or **CC-B** 2D barcode component to create the EAN.UCC composite symbols. In such case, only the Barcode, Data (only for Delphi/C++ Builder 2009 or later),AutoCheckDigit, Height, DisplayText, TextPosition, TextAlignment, TextFont, TextVSpacing, TextHSpacing, ExtraFontSize, LeftQuietZoneSpacing, and the RightQuietZoneSpacing properties are useful, the value of other properties will be ignored and they will be set automatically depending on the settings of the 2D barcode component. In addition, the Height property should be set to a value larger than zero, and the Image should not be normally set.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 7 characters, othwise 8 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- ShowQuietZoneMark
- LeftQuietZoneSpacing
- RightQuietZoneSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.26 TBarcode1D_FIM

The component is used to create the Facing Identification Mark (FIM) barcode symbol. It's defined in the pFIM unit.

The Facing Identification Mark (FIM) is a barcode symbology designed by the United States Postal Service to assist in the automated processing of mail. It allows the proper facing of mail for cancellation. It also identifies the manner in which postage is paid (e.g., business reply mail or IBI - Information Based Indiciapostage) and whether that business reply mail has a POSTNET barcode.

The following four codes are in use:

- **FIM A:** It is used for courtesy reply mail and metered reply mail with a preprinted POSTNET bar code. In both of these types of mail, the postage is prepaid, either by a postage stamp in the case of courtesy reply mail or by a postage meter in the case of metered reply mail.
- **FIM B:** It is used for business reply mail without a preprinted ZIP + 4 bar code.
- **FIM C:** It is used for business reply mail with a preprinted ZIP + 4 bar code.
- **FIM D:** It is used only with IBI postage.

Properties:

- Image
- FIMType
- Data (*)
- Module
- Height
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextHSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

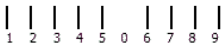
- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.27 TBarcode1D_Flattermarken



The component is used to create the Flattermarken barcode symbol. It's defined in the pFlattermarken unit.

The Flattermarken barcode is a special "barcode" used for recognizing the correct sequence of pages in print-shops.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Variable
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextHSpacing
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.28 TBarcode1D_IATA

The component is used to create the IATA barcode symbol. It's defined in the pIATA unit.

The IATA barcode symbology is also known as IATA 2 of 5, 2 of 5 IATA, 2/5 IATA, International Air Transport Assosiation 2 of 5, etc. It is based on Industrial 2 of 5 standard, and it is a low-density variable-length numeric-only barcode symbology. It is used by International Air Transport Assosiation (IATA) for the management of air cargo.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Variable
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.29 TBarcode1D_Identcode

The component is used to create the Identcode barcode symbol. It's defined in the pIdentcode unit.

This Identcode barcode symbology is also known as DPI, Deutsche Post AG IdentCode, German Postal 2 of 5 IdentCode, Deutsche Frachtpost IdentCode IdentCode, Deutsche Post AG DHL. It is used by German Post (Deutsche Post AG, Deutsche Frachtpost). The barcode contains a tracking number providing a identification of the customer (sender) and the mail piece. The value to encode length is fixed to 12 digits. The value to encode must have the following structure:

1. 2 digits for ID of primary distribution center.
2. 3 digits for Customer ID.
3. 6 digits for Mailing number.
4. 1 digit for check digit.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 11 characters, othwise 12 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- SplitText
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.30 TBarcode1D_ITF14

The component is used to create the ITF-14 barcode symbol. It's defined in the pITF14 unit.

This ITF-14 barcode symbology is also known as UPC Shipping Container Symbol ITF-14, ITF14, Case Code, UPC Case Code, EAN/UCC-14, EAN-UCC-14, DUN-14, GTIN-14, UCC-12, EAN/UCC-13, ITF, etc. It is used to mark cartons, cases, or pallets that contain products which have a UPC or EA product identification number. The value to encode must have the following structure:

1. 1 digit for Packaging indicator.
2. 2 digits for UPC numbering system or EAN prefix.
3. 5 digits for Manufacturer identification number.
4. 5 digits for Item identification number.
5. 1 digit for Checksum, If the AutoCheckDigit property is set to true, it doesn't need to be entered in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, and it will be automatically calculated.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 13 characters, othwise 14 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- SplitText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- LeftSpacing
- RightSpacing
- BearerBars
- BearerWidth
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.31 TBarcode1D_ITF16

The component is used to create the ITF-16 barcode symbol. It's defined in the pITF16 unit.

This ITF-16 barcode symbology is also known as UPC Shipping Container Symbol ITF-16, ITF16, etc. It is used to mark cartons, cases, or pallets tha contain products which have a UPC or EAN product identification number, in Japan. The value to encode must have the following structure:

1. 1 digit fixed.
2. 2 digit for Packaging indicator.
3. 2 digits for UPC numbering system or EAN prefix.
4. 5 digits for Manufacturer identification number.
5. 5 digits for Item identification number.
6. 1 digit for Checksum, If the AutoCheckDigit property is set to true, it doesn't need to be entered in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, and it will be automatically calculated.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 15 characters, othwise 16 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- SplitText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- LeftSpacing
- RightSpacing
- BearerBars
- BearerWidth
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.32 TBarcode1D_ITF6

The component is used to create the ITF-6 barcode symbol. It's defined in the pITF6 unit.

This ITF-6 barcode symbology is also known as UPC Shipping Container Symbol ITF-6, ITF6, etc. It is a shortened version of ITF-14 and is specified by GS1.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. 6 digits.
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- SplitText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- LeftSpacing
- RightSpacing
- BearerBars
- BearerWidth
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.33 TBarcode1D_JapanPost

The component is used to create the Japan Postal 4-State barcode symbol. It's defined in the pJapanPost unit.

The Japan Postal 4-State barcode is a clocked barcode similar in appearance to 4 State code, with a mod 19 checkdigit. It will accept digits and uppercase letters and the hyphen. The data consists of a 7 digit postal code plus address data.

Technical Details:

The formats of the Japan Postal 4-State barcode are specified in the following:

- **Postcode:** Length: 6 numbers; Characters set: "0"-"9"; The sectionmay have a hyphen between the 3th and 4th number (eg. 123-4567) and this hyphen does not appear in the encoded data.
- **Address data:** Length: variable, maximum 13 code words, every number or hyphen use 1 code word, and every upper letter use 2 code words. If the address data is less than 13 code words the remaining code words are filled with control characters to make the length 13; Characters set: "0"-"9", "-", "A"-"Z". Note, the hyphens in the address data are encoded.

Example: 154-0023-1-3-2-A-507, 1540023-1-3-2-A-507

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- SplitText
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.34 TBarcode1D_KIX4S

The component is used to create the Royal TPG Post KIX 4-State (KIX4S) barcode symbol. It's defined in the pKIX4S unit.

The KIX 4-State barcode symbology is known as Kix Barcode, Klantenindex (client index) Barcode, Dutch KIX 4-State Bar Code, Dutch KIX TPG KIX, and TPGPOST KIX. It is used by Royal Dutch TPG Post (TNT Post Group, Netherlands) for Postal code and automatic mail sorting. It provides information about the address the receiver. It encodes alpha-numeric characters ("0"-"9", "A"-"Z").

Technical Details:

The formats of the KIX 4-State barcode are specified in the following:

- **Postcode:** Length: 6 characters; Characters set: "0"-"9", "A"-"Z"; It consists of 4 numbers ("0"-"9") then 2 letters ("A"-"Z").
- **House, Mailbox or Freephone number:** Length: variable, maximum 5 characters; Characters set: "0"-"9".
- **Separation stabbing:** Length: 1 characters; Characters set: "X"; It is required if the Additions number exists.
- **Additions number:** Optional; Length: variable, maximum 6 characters; Characters set: "0"-"9", "A"-"Z".

Example: 2130VA80430

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.35 TBarcode1D_Leitcode

The component is used to create the Leitcode barcode symbol. It's defined in the pLeitcode unit.

This Leitcode barcode symbology is also known as DPL, Deutsche Post Leitcode, German Postal 2 of 5 LeitCode, LeitCode, CodeLeitcode, Deutsche Post A(DHL. It is used by German Post (Deutsche Post AG, Deutsche Frachtpost). The barcode gives an indication of the destination. The value to encode length is fixed to 14 digits. The value to encode must have the following structure:

1. 5 digits for Postal Code (Postleitzahl, PLZ).
2. 3 digits for Street ID/number.
3. 3 digits for House number.
4. 2 digits for Product code.
5. 1 digit for check digit.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 13 characters, othwise 14 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- SplitText
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.36 TBarcode1D_MSI

The component is used to create the MSI barcode symbol. It's defined in the pMSI unit.

The MSI barcode symbology is also known as MSI/Plessey, Modified Plessey, etc. It is a variable length, numeric-only continuous symbology. It was developed by the MSI Data Corporation, based on the original Plessey Code. It is used primarily to mark retail shelves for inventory control.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Variable
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- CheckMethod
- Mod11Weighting
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.37 TBarcode1D_OneCode

The component is used to create the OneCode barcode symbol. It's defined in the pOneCode unit.

The OneCode barcode symbology is also known as Intelligent Mail barcode, IMB, 4-State Barcode, USPS OneCode Solution o USPS 4-State Customer Barcode (abbreviated 4CB, 4-CB, or USPS4CB). It is a 65-bar code for use on mail in the United States. It combines routing ZIP Code information and tracking information into a single 4-state code. It effectively encodes data from POSTNET and PLANET barcodes into a single barcode while providing a greater range of tracking data. The hope is that it may provide information and benefits to both mailers and postal officials.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed(20, 25, 29, or 31 characters)
- **Code type:** Clocked 1D Continuous
- **Example:** Tracking: 20702123456789012345, Routing: 12345678901

Properties:

- Image
- Tracking
- Routing
- Data (*)
- Module
- Height
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- SplitText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.38 TBarcode1D_Patch

The component is used to create the Patch Code. It's defined in the pPatch unit.

A Patch Code is a pattern of parallel, alternating black bars and spaces that is printed on a document. Kodak Scanners which have Patch Code reading capability can recognize patch documents and automatically assign a document image level, increment the document image address, or perform Color on the Fly functionality.

The following six codes are in use:

- **Patch 1:** Patch Type 1 can be used by the host for post-scan image control for the i800/i1800 (with image addressing) Series Scanners (they are not used for image addressing).
- **Patch 2:** Patch Type 2 can be used to assigns image level 2 to the current document.
- **Patch 3:** Patch Type 3 can be used to assigns image level 3 to the current document.
- **Patch 4 / Toggle Patch:** Patch Type 4 can be used by the host for post-scan image control for the i800/i1800 (with image addressing) Series Scanners (they are not used for image addressing). The Toggle Patch may be used to switch back and forth from bi-tonal and color/grayscale scanning for the i280, 3590C, i600, i800 and i1800 (without image addressing) Series Scanners. This provides Color on the Fly during capture, with no need for post-scan processing by the host application.
- **Patch 6:** Patch Type 6 can be used by the host for post-scan image control for the i800/i1800 (with image addressing) Series Scanners (they are not used for image addressing).
- **Patch T / Transfer Patch:** Patch Type T can be used to assigns a predefined image level to the next document. The predefined image level is based upon the transfer patch definition which is defined for each application. For example, if the transfer patch definition is image level 2, then use of a transfer patch assigns image level 2 to the next document.

Properties:

- Image
- PatchType
- Data (*)
- Module
- Height
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextHSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.39 TBarcode1D_PharmacodeOneTrack

The component is used to create the One-Track Pharmacode barcode symbol. It's defined in the pPharmacodeOneTrack unit.

The One-Track Pharmacode barcode symbology is known as Pharmacode Laetus, Pharmacode one-track, Pharmacode 1-track, 1-track Pharmacode, Pharmaceutical Binary Code. It was developed by Laetus. One-Track Pharmacodeis used in the pharmaceutical industry as a packing control system. The Pharmacode barcode is found extensively on the packaging of pharmaceutical products. Pharmacode can represent only a single integer from 3 to 131070.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Variable. It can represent only a single integer from 3 to 131070.
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- BarColor
- BarColors
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextHSpacing
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.40 TBarcode1D_PharmacodeTwoTrack

The component is used to create the Two-Track Pharmacode barcode symbol. It's defined in the pPharmacodeTwoTrack unit.

The Two-Track Pharmacode barcode symbology is known as Pharmacode two-track, 2-track Pharmacode, Pharmacode 2-track. It was developed by Laetus. Two-Track Pharmacodeis used in the pharmaceutical industry as a packing control system. It can represent only a single integer from 3 to 64570080.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Variable. It can represent only a single integer from 3 to 64570080.
- **Code type:** Clocked 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- BarColor
- BarColors
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextHSpacing
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.41 TBarcode1D_Planet

The component is used to create the Postal Alpha Numeric Encoding Technique (PLANET) barcode symbol. It's defined in the pPLANET unit.

The Postal Alpha Numeric Encoding Technique (PLANET) barcode symbology is used by the United States Postal Service to identify and track pieces of mail during delivery - the Post Office's "CONFIRM" services. The PLANET barcode is either 12 or 14 digits long. The first two digits of the barcode identify the particular Confirm service you are using, including the Destination Confirm and the Origin Confirm. For the Origin Confirm barcode, the next is the Customer ID, it is a 9- or 11-digit ID defined by the mailer and is used to identify the mailpiece. For the Destination Confirm barcode, the next is a Subscriber ID and a Mailing ID. The Subscriber ID is an unique 5-digit ID assigned by the Postal Service to identify mailers using Confirm service. The Mailing ID is a 4- or 6-digit field defined by the mailer to identify the actual mailing. The last one digit is the checksum digit.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 11, or 13 characters, othwise 12, or 14 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.42 TBarcode1D_Plessey

The component is used to create the Plessey barcode symbol. It's defined in the pPlessey unit.

The Plessey barcode symbology is known as Plessey Bidirectional. It is an older code still popular in some industries, and it is a lower-density variable-length continuous barcode symbology, It was developed by the Plessey Company in England with formal specifications first dated March 1971. A variation of Plessey Code is known as Anker Code. It has inverted CRC. Anker Code was used in European point of sale systems prior to the advent of EAN.

Technical Details:

- **Characters set:** 0123456789ABCDEF
- **Length:** Variable
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- Bidirectional
- UKMode
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.43 TBarcode1D_PostBar

The component is used to create the PostBar barcode symbol. It's defined in the pPostBar unit.

The PostBar barcode symbology is also known as CPC 4-State. It is the black-ink barcode system used by Canada Post in its automated mail sorting and delivery operations. Canada Post uses nine different formats of PostBar codes - three "domestic" barcodes, used on mail within Canada, two "global" codes, used to route mail outside Canada, three "service" codes, used on customer-applied barcodes, and an "internal" code, used for testing, maintenance, and tracking purposes by Canada.

Technical Details:

The formats of the PostBar codes are specified in the following:

- **Domestic**
  - **Format D07:** Z ANANAN
    - Z: Data Content Identifier (Character "A")
    - ANANAN: Postal Code (Length: 6 characters; Characters set: A: "0"-"9"; N: "A"-"Z")

    Example: AK1A7R8
  - **Format D12:** Z ANANAN ZZZZ
    - Z: Data Content Identifier (Character "B")
    - ANANAN: Postal Code (Length: 6 characters; Characters set: A: "0"-"9"; N: "A"-"Z")
    - ZZZZ: Address Information (Length: 4 characters; Characters set: [SPACE], "0"..."9", "A"-"Z")

    Example: BK1A4S21234
  - **Format D22:** Z ANANAN ZZZZ ZZZZZZZZZZ
    - Z: Data Content Identifier (Character "C")
    - ANANAN: Postal Code (Length: 6 characters; Characters set: A: "0"-"9"; N: "A"-"Z")
    - ZZZZ: Address Information (Length: 4 characters; Characters set: [SPACE], "0"..."9", "A"-"Z")
    - ZZZZZZZZZZ: Customer Information (Length: 0-11 characters; Characters set: [SPACE], "0"-"9", "A"-"Z")

    Example: CL3B4T91420CFFMIPLXF6V

- **Global**
  - **Format G12:** Z NNN ZZZZZZZZ
    - Z: Data Content Identifier (Character "1")
    - NNN: Country Code (Length: 3 characters; Characters set: "0"-"9")
    - ZZZZZZZZ: Postal Code (Length: 8 characters; Characters set: [SPACE], "0"..."9", "A"-"Z")

    Example: 118091266ABC
  - **Format G22:** Z NNN ZZZZZZZZ ZZZZZZZZZZ
    - Z: Data Content Identifier (Character "2")
    - NNN: Country Code (Length: 3 characters; Characters set: "0"-"9")
    - ZZZZZZZZ: Postal Code (Length: 8 characters; Characters set: [SPACE], "0"..."9", "A"-"Z")
    - ZZZZZZZZZZ: Customer Information (Length: 1-10 characters; Characters set: [SPACE], "0"..."9", "A"-"Z")

    Example: 2216HA97PPABFFMIPL659V
- **Service**
  - **Format S06:** Z Z ZZZZZZZZZZZZZZZZZZZZ
    - Z: Data Content Identifier (Character "S")
    - Z: Bar Code Sequencer (Length: 1 character; Characters set: [SPACE], "0"..."9", "A"-"Z")
    - ZZZZZZZZZZZZZZZZZZZ: Service Information (Length: 0-19 characters; Characters set: [SPACE], "0"..."9", "A"-"Z")

    Example: S0ABCDEFGHIJ123456789
  - **Format S11:** Z ZZZZZZZZZZ
    - Z: Data Content Identifier (Character "T")
    - ZZZZZZZZZZ: Service Information (Length: 0-10 characters; Characters set: [SPACE], "0"..."9", "A"-"Z")

    Example: TABCDE12345
  - **Format S21:** Z ZZZZZ
    - Z: Data Content Identifier (Character "U")
    - ZZZZZ: Service Information (Length: 0-5 characters; Characters set: [SPACE], "0"..."9", "A"-"Z")

    Example: UABC12
- **Internal use**
  - **Format C10:** Z ANANAN BBB
    - Z: Data Content Identifier (Character "Z")
    - ANANAN: Postal Code (Length: 6 characters; Characters set: A: "0"-"9"; N: "A"-"Z")
    - BBB: Machine Identifier (Length: 0-3 characters; Characters set: "0"..."9")

    Example: ZK1A0B1097

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- SplitText
- HideCheckDigitsText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.44 TBarcode1D_PostNet

The component is used to create the USPS PostNet barcode symbol. It's defined in the pPostNet unit.

The USPS Postnet barcode symbology is known as PostNet. It is a numeric symbology. It was invented by US Postal Office to encode ZIP information. The US delivery address coding can be of three forms (1) 5-digit ZIP; (2) 5-digit ZIP + 4 code; (3) 11-digit delivery point code.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 5, 9, or 11 characters, othwise 6, 10, or 12 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.45 TBarcode1D_PZN

The component is used to create the PZN barcode symbol. It's defined in the pPZN unit.

The PZN barcode symbology is also known as Pharma-Zentral-Nummer, Pharmazentralnummer, Code PZN, CodePZN, Pharma Zentral Nummer, etc. It is used for distribution of pharmaceutical / health care products in Germany. It has the following structure:

1. The "PZN" characters which is not really encoded, It does not need to be entered in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property.
2. 6 digits for Pharmacode.
3. 1 digit for check digit, If the AutoCheckDigit property is set to true, it doesn't need to be entered in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, and it will be automatically calculated.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 6 characters, othwise 7 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- ShowGuards
- InterGap
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.46 TBarcode1D_RM4SCC

The component is used to create the Royal Mail 4-state Customer Code (RM4SCC) barcodes symbol. It's defined in the pRM4SCC unit.

The RM4SCC barcode symbology is known as CBC (Customer Bar Code) within Royal Mail. It is a height-modulated barcode symbology and it can be used with Royal Mail's Cleanmail system, as an alternative to OCR readable fonts, to allow businesses to easily and cheaply send large quantities of letters. In Singapore, it is known as Singapore 4-State Postal Code barcode, Singapore 4-State Postal, SingPost 4-State, SingPost Barcode, and Singapore 4-State Code. and it is used by Singapore Pos (SingPost) for Postal code and automatic mail sorting.

Technical Details:

The formats of the RM4SCC codes are specified in the following:

- **Postcode:** Length: 5, 6, or 7 characters; Characters set: "0"-"9", "A"-"Z"
- **Delivery Point Suffix (DPS):** Length: 2 characters; Characters set: "0"-"9", "A"-"Z"; A DPS consists of a number ("1"-"9") then a letter("A", "B", "D"-"H", "J", "L", "N", "P"-"U", "W"-"Z"). The number can be any from 1 to 9 but not 0. The letter can be any except C, I, K, M, O or V.
- **Check digits:** Length: 1 characters; Characters set: "0"-"9", "A"-"Z". The checksum character is used as a means of error detection toensure that the rest of the barcode is correct. It will be automatically appended if the AutoCheckDigit property is set to true.

Example: SN34RD1A

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.47 TBarcode1D_Telepen

The component is used to create the Telepen barcode symbol, including the Telepen ASCII, Telepen Numeric, and Telepen Full ASCII. It's defined in the pTelepen unit.

The Telepen barcode symbology is also known as Telepen Barcode, Telepen Numeric, Telepen Full ASCII, SB Electronic Systems Barcode, etc. It can be used to represent the full range of ASCII characters (ASCII 00 - ASCII 127). It can also be used to represent numeric data in double-density mode. Telepen systems have been implemented in many countries and very widely in the UK. Most Universities and other academic libraries use Telepen, as do many public libraries. There are 3 encode modes for the Telepen system:

- **Full ASCII mode:** Encodes all ASCII characters (ASCII 0 - ASCII 127).
- **Numeric mode:** Encodes numeric data in double-density mode, An ASCII 16 character will switch to the ASCII mode.
- **ASCII mode:** Encodes all ASCII characters, an ASCII 16 character will switch to the Numeric mode.

Technical Details:

- **Characters set:** Numeric mode: 0123456789; ASCII and Full ASCII mode: All 128 ASCII characters (ASCII 0 - ASCII 127)
- **Length:** Variable
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- Mode
- Guards
- ExtraChar
- OddEncode
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Size
- CopyToClipboard
- Draw
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.48 TBarcode1D_UPCA

The component is used to create the UPC-A barcode symbol. It's defined in the pUPCA unit.

The UPC-A barcode symbology is also known as Universal Product Code version A, UPC Code, UPC Symbol, GTIN-12, GS1-12, UCC-12. It is used for marking products which are sold at retail in the USA. The barcode identifies the manufacturer and specific product so point-of- sale cash register systems can automatically look up the price. The UPC-A Code and the assignment of manufacturer ID numbers is controlled in the USA. by the Uniform Code Council (UCC).

The value to encode by UPC-A has the following structure:

1. 1 digit for Number System (0: Regular UPC codes, 1: Reserved, 2: Random weight items marked at the store, 3: National Drug Code and National Health Related Items code, 4 No format restrictions, for in-store use on non-food items, 5: For use on coupons, 6: Reserved, 7: Regular UPC codes, 8: Reserved, 9: Reserved).
2. 5 digits for Manufacturer (Company) Code or prefix. This number is assigned by the Uniform Code Council (UCC).
3. 5 digits for Product Code which is assigned by the manufacturer.
4. 1 digit for check digit.

An EAN-2 or an EAN-5 barcode may be added.

The UPC-A barcode symbol can be used together with a 4-column CC-A or a 4-column CC-B 2D symbol to create the EAN.UCC composite symbol. If you use the **2D Barcode VCL Components** package, the component can be used as the value of the **Liner** property of the **CC-A** or **CC-B** 2D barcode component to create the EAN.UCC composite symbols. In such case, only Barcode, Data (only for Delphi/C++ Builder 2009 or later),AutoCheckDigit, Height, DisplayText, TextPosition, TextAlignment, TextFont, TextVSpacing, TextHSpacing, ExtraFontSize, LeftQuietZoneSpacing, and RightQuietZoneSpacing properties are useful, the value of other properties will be ignored and they will be set automatically depending on the settings of the 2D barcode component. In addition, the Height property should be set to a value larger than zero, and the Image should not be normally set.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 11 characters, othwise 12 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- ExtraFontSize
- TextCSpacing
- TextVSpacing
- TextHSpacing
- LeftQuietZoneSpacing
- RightQuietZoneSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.49 TBarcode1D_UPCE

The component is used to create the UPC-E barcode symbol, including the UPC-E0 and UPC-E1. It's defined in the pUPCE unit.

The UPC-E barcode symbology is also known as Universal Product Code version E, UPC-E0, E0, UPC-E1, E1, GTIN-12 with lead "0", GS1-12, UCC-12, etc. It is variation of UPC-A which allows for a more compact barcode by eliminating "extra" zeros. Since the resulting UPC-E barcode is about half the size as anUPC-A barcode, UPC-E is generally used on products with very small packaging where a fullUPC-A barcode couldn't reasonably fit. An UPC-E barcode represents 7 digits and a check digit, and the first digit (number system) is 0 (UPC-E0) or 1 (UPC-E1).

An EAN-2 or an EAN-5 barcode may be added.

The UPC-E barcode symbol can be used together with a 2-column CC-A or a 2-column CC-B 2D symbol to create the EAN.UCC composite symbol. If you use the **2D Barcode VCL Components** package, the component can be used as the value of the **Liner** property of the **CC-A** or **CC-B** 2D barcode component to create the EAN.UCC composite symbols. In such case, only Barcode, Data (only for Delphi/C++ Builder 2009 or later),AutoCheckDigit, Height, DisplayText, TextPosition, TextAlignment, TextFont, TextVSpacing, TextHSpacing, ExtraFontSize, LeftQuietZoneSpacing, and RightQuietZoneSpacing properties are useful, the value of other properties will be ignored and they will be set automatically depending on the settings of the 2D barcode component. In addition, the Height property should be set to a value larger than zero, and the Image should not be normally set.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 7 characters, othwise 8 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- ExtraFontSize
- TextCSpacing
- TextVSpacing
- TextHSpacing
- LeftQuietZoneSpacing
- RightQuietZoneSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.50 TBarcode1D_UPCE0

The component is used to create the UPC-E0 barcode symbol. It's defined in the pUPCE0 unit.

The UPC-E0 barcode symbology is also known as Universal Product Code version E, UPC-E0, E0, GTIN-12 with lead "0", GS1-12, UCC-12, etc. It is a variation of UPC-A which allows for a more compact barcode by eliminating "extra" zeros. Since the resulting UPC-E0 barcode is about half the size as an UPC-A barcode, UPC-E0 is generally used on products with very small packaging where a full UPC-A barcode couldn't reasonably fit. An UPC-E0 barcode represents 6 digits with an implied number system 0, and a check digit. The number system 0 (first digit) is not required, it will be inserted automatically.

An EAN-2 or an EAN-5 barcode may be added.

The UPC-E0 barcode symbol can be used together with a 2-column CC-A or a 2-column CC-B 2D symbol to create the EAN.UCC composite symbol. If you use the **2D Barcode VCL Components** package, the component can be used as the value of the **Liner** property of the **CC-A** or **CC-B** 2D barcode component to create the EAN.UCC composite symbols. In such case, only Barcode, Data (only for Delphi/C++ Builder 2009 or later), AutoCheckDigit, Height, DisplayText, TextPosition, TextAlignment, TextFont, TextVSpacing, TextHSpacing, ExtraFontSize, LeftQuietZoneSpacing, and RightQuietZoneSpacing properties are useful, the value of other properties will be ignored and they will be set automatically depending on the settings of the 2D barcode component. In addition, the Height property should be set to a value larger than zero, and the Image should not be normally set.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 6 characters, othwise 7 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- ExtraFontSize
- TextCSpacing
- TextVSpacing
- TextHSpacing
- LeftQuietZoneSpacing
- RightQuietZoneSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.51 TBarcode1D_UPCE1

The component is used to create the UPC-E1 barcode symbol. It's defined in the pUPCE1 unit.

The UPC-E1 barcode symbology is also known as Universal Product Code version E1, UPC-E1, E1, GTIN-12 with lead "1", etc. It is a variation of UPC-A which allows for a more compact barcode by eliminating "extra" zeros. Since the resulting UPC-E1 barcode is about half the size as an UPC-A barcode, UPC-E1 is generally used on products with very small packaging where a full UPC-A barcode couldn't reasonably fit. An UPC-E1 barcode represents 6 digits with an implied number system 1, and a check digit. The number system 1 (first digit) is not required, it will be inserted automatically.

An EAN-2 or an EAN-5 barcode may be added.

The UPC-E1 barcode symbol can be used together with a 2-column CC-A or a 2-column CC-B 2D symbol to create the EAN.UCC composite symbol. If you use the **2D Barcode VCL Components** package, the component can be used as the value of the **Liner** property of the **CC-A** or **CC-B** 2D barcode component to create the EAN.UCC composite symbols. In such case, only Barcode, Data (only for Delphi/C++ Builder 2009 or later), AutoCheckDigit, Height, DisplayText, TextPosition, TextAlignment, TextFont, TextVSpacing, TextHSpacing, ExtraFontSize, LeftQuietZoneSpacing, and RightQuietZoneSpacing properties are useful, the value of other properties will be ignored and they will be set automatically depending on the settings of the 2D barcode component. In addition, the Height property should be set to a value larger than zero, and the Image should not be normally set.

Technical Details:

- **Characters set:** 0123456789
- **Length:** Fixed. If the property AutoCheckDigit is set to true, it's 6 characters, othwise 7 characters (the check digit can be specified by you)
- **Code type:** Linear 1D Continuous

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- ExtraFontSize
- TextCSpacing
- TextVSpacing
- TextHSpacing
- LeftQuietZoneSpacing
- RightQuietZoneSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.1.52 TBarcode1D_UPU

The component is used to create the UPU barcode symbol. It's defined in the pUPU unit.

The UPU barcode symbology is used by the barcode supported system of tracking, which is able to inform the sender as well as the receiver permanently about the actual position of their consignment. For that all consignments will be provided with an UPU linear barcode. These barcodes will be each scanned from taking over the consignment at each change of means of transportation. The datas will be passed on to the relevant computers by means of telecommunication. The permanent data-input creates a curriculum vitae for each consignement questionable from customers.

It has the following structure:

1. **Service Indicators**: 2 continuous digits or combination of letters, service indicators (prefixes) for internal services. Countries are free to use whatever service indicators (prefixes) they wish for internal service.
2. **Track Code**: 10 continuous digits. Countries are free to use.
3. **Checking Digit** 1 digit for checking digit, If the AutoCheckDigit property is set to true, it can be set to a "?", or a digit from "0" to "9". The checking digit will be calculated automatically, and replace the dight you set in the barcode. If the AutoCheckDigit property is set to false, it can be only set to a dignt from "0" to "9".
4. **Country Code**: Consists of two capital letters, corresponding ISO-standard " ISO 3166".

Technical Details:

- **Characters set:** "0"-"9", "A"-"Z", "-", ".", "$", "/", "+", "%", and " " (space)
- **Length:** Fixed (13 characters)
- **Code type:** Linear 1D Discrete

Properties:

- Image
- Barcode
- Data (*)
- Module
- Height
- Ratio
- AutoCheckDigit
- ShowGuards
- InterGap
- SplitText
- BarColor
- SpaceColor
- Orientation
- LeftMargin
- TopMargin
- BarcodeWidth
- BarcodeHeight
- Stretch
- StretchTextHeight
- DisplayText
- TextPosition
- TextAlignment
- TextFont
- TextCSpacing
- TextVSpacing
- TextHSpacing
- FullEncoded
- Locked

Methods:

- Create
- Destroy
- Assign
- Clear
- Draw
- Size
- CopyToClipboard
- DrawTo
- DrawToSize
- Print
- PrintSize

Events:

- OnChange
- OnInvalidChar
- OnInvalidLength
- OnInvalidDataChar (*)
- OnInvalidDataLength (*)
- OnDrawText
- OnDrawBarcode

(*): The Data property, OnInvalidDataLength and OnInvalidDataChar events are available only for the Delphi/C++ Builder 2009 or later.

4.2 TBarcodeCustomParameters

It contains fields to specify the parameters (e.g. position, size, etc.) for the barcode symbol. It is defined in the pBarcode1D unit.

The record is used in the OnDrawBarcode and OnDrawText events.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TDisplayText = (dtNone, dtBarcode, dtFullEncoded);
  { Defined in the pCore1D unit }
  TTextPosition = (tpTopIn, tpTopOut, tpBottomIn, tpBottomOut);
  { Defined in the pCore1D unit }
  TTextAlignment = (taLeft, taCenter, taRight, taJustify, taLeftQuietZone, taCenterQuietZone, taRightQuietZone,
    taJustifyQuietZone, taCustom);
  { Defined in the pBarcode1D unit }
  TBarcodeCustomParameters = record
    Alpha: Double;
    Origin: TPoint;
    Offset: TPoint;
    DensityRate: Double;
    FullEncoded: string;
    Text: string;
    DisplayText: TDisplayText;
    TextPosition: TTextPosition;
    TextAlignment: TTextAlignment;
    TextFont: TFont;
    ExtraFontSize: Integer;
    LeftQuietZone_Spacing: Integer;
    RightQuietZone_Spacing: Integer;
```

```
    LeftQuietZone_Width: Integer;
    RightQuietZone_Width: Integer;
    LeftQuietText_Height: Integer;
    RightQuietText_Height: Integer;
    Symbol_Width: Integer;
    Symbol_Height: Integer;
    Symbol_V_Offset: Integer;
    Symbol_H_Offset: Integer;
    Total_Left: Integer;
    Total_Top: Integer;
    Total_Width: Integer;
    Total_Height: Integer;
  end;
```

Fields:

- **Alpha:** Double; The angle in radian that the barcode symbol will be rotated. See diageam:

- **Origin:** TPoint; The coordinate of the upper-left corner of the barcode symbol after the barcode symbol is rotated. The X value is in logical dots or pixels in the horizontal direction. The Y value is in logical dots or pixels in the vertical direction. See diageam:

- **Offset:** TPoint; For TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_UPCA, TBarcode1D_EAN8, TBarcode1D_EAN13, or TBarcode1D_EAN128 barcode components, if it is used as the liner component in the EAN.UCC composite barcode symbols (if you use the **2D Barcode VCL Components** package, it is used as the value of **Liner** property of the **CC-A**, **CC-B** or **CC-C** 2D barcode component), the value of this field is the coordinate offset, in logical pixels (Draw, DrawTo) or dots (Print) in the horizontal direction, for the upper-left corner of the barcode symbol. The offset is relative to the upper-left corner of entire EAN.UCC composite barcode symbol. For other barcode symbols, the value of this field is (0, 0). See diageam:

- **DensityRate:** Double; It is the ratio of the horizontal logical DPI and veritical logical DPI of the canvas.

- **FullEncoded:** string; Contains the barcode text and the check digit that's automatically appended to the barcode symbol. The start code and the stop code aren't included.

  For the TBarcode1D_OneCode component, it includes all the tracking and routing, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

  See also the "FullEncoded" property.

- **Text:** String; The text which will be represented as the human readable text. It's the value of the FullEncoded property, or the value of the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, or an empty string, base on the value of the DisplayText property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the full encoded value generated from the Barcode or Data (only for Delphi/C++ Builder 2009 or later) parmaters, or the value of the Barcode or Data (only for Delphi/C++ Builder 2009 or later) parameter, or an empty string, base on the value of the DisplayText field in the BarcodeTextDefine parameter).

  If the human readable text is represented, for the TBarcode1D_OneCode barcode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

  For the TBarcode1D_Code128 and TBarcode1D_EAB128 components, you can encode a block of binary (bytes) data into the barcode symbol. In this case, you can use the OnDecodeText event to decode the text from the block of binary (bytes) data in order to output it as the barcode text into the barcode symbol, and the field is equal to the BarcodeText parameter of the OnDecodeText event function.

  For the TBarcode1D_FIM component, it is single character that denotes the FIM type, form "A" to "E", the "E" character dnotes an empty barcode symbol.

  For the TBarcode1D_Patch component, it is single character that denotes the Ptach type, including "0", "1", "2", "3", "4", "6", and "T", the "0" character dnotes an empty barcode symbol.

- **DisplayText**: TDisplayText; Current value of the DisplayText property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the value of the DisplayText field of the BarcodeTextDefine parameter). It specifies whether to represent the human readable text and what will display as the human readable text. This field can have one of following values (defined in the pCore1D unit):

  - **dtNone:** Don't represent the human readable text.
  - **dtBarcode:** Represent the barcode text that is specified in the Barcode parameter.
  - **dtFullEncoded:** Represent the barcode text that is specified in the Barcode parameter, and the check digit that's automatically appended to the barcode symbol.

See also the "DisplayText" property.

- **TextPosition:** TTextPosition; Current value of the TextPosition property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the value of the TextPosition field of the BarcodeTextDefine parameter). It specifies the position of the human readable text (specifies the vertical alignment of the human readable text within the barcode symbol).

  This field can have one of these values (defined in the pCore1D unit):

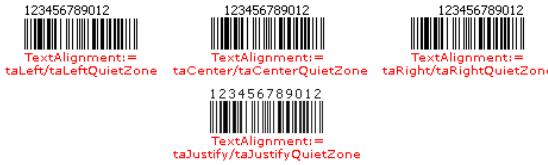
  - **tpTopIn:** Justifies the human readable text to the top in the barcode symbol, the bars and spaces on both left and right sides of the human readable text will be reserved. See diagram:

    If the TextAlignment field is set to taJustify or taJustifyQuietZone, it is the same as using the tpTopOut.

    For TBarcode1D_ITF6, TBarcode1D_ITF14, TBarcode1D_ITF16, TBarcode1D_PLANET, TBarcode1D_PostNet, TBarcode1D_JapanPost, TBarcode1D_AP4SC, TBarcode1D_KIX4S, TBarcode1D_RM4SCC, TBarcode1D_PharmacodeTwoTrack, TBarcode1D_PostBar, and TBarcode1D_OneCode barcode components, it is the same as using the tpTopOut.

    For TBarcode1D_EAN2 and TBarcode1D_EAN5 barcode components, if the TextAlignment field is set to taCustom, it is the same as using the tpTopOut.

  - **tpTopOut:** Justifies the human readable text to the top in the barcode symbol, the bars and spaces on both left and right sides of the human readable text will be erased. See diagram:

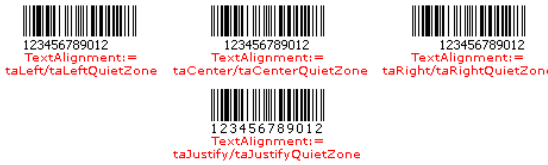
  - **tpBottomIn:** Justifies the human readable text to the bottom in the barcode symbol, the bars and spaces on both left and right sides of the human readable text will be reserved.

    If the TextAlignment field is set to taJustify or taJustifyQuietZone, it is the same as using the tpBottomOut.

    For TBarcodeD_ITF6, TBarcodeD_ITF14, TBarcodeD_ITF16, TBarcodeD_PLANET, TBarcodeD_PostNet, TBarcodeD_JapanPost, TBarcodeD_AP4SC, TBarcodeD_KIX4S, TBarcodeD_RM4SCC, TBarcodeD_PharmacodeTwoTrack, TBarcodeD_PostBar, and TBarcodeD_OneCode barcode components, it is the same as using the tpBottomOut.

    For TBarcode1D_EAN2 and TBarcode1D_EAN5 barcode components, if the TextAlignment field is set to taCustom, it is the same as using the tpBottomOut. See diagram:

  - **tpBottomOut:** Justifies the human readable text to the bottom in the barcode symbol, the bars and spaces on both left and right sides of the human readable text will be erased. See diagram:

  See also the "TextPosition" property.

- **TextAlignment:** TTextAlignment; Current value of the TextAlignment property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the value of the TextAlignment of the BarcodeTextDefine parameter). It determines the horizontal alignment the human readable text within the barcode symbol.

  This field can have one of these values (defined in the pCore1D unit):

  - **taLeft:** Aligns the human readable text to the left within the barcode symbol. See diagram:

    For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren't included when align the human readable text. See diagram:

  - **taCenter:** Centers the human readable text horizontally within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren't included when align the human readable text. See diagram:

- **taRight:** Aligns the human readable text to the right within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren't included when align the human readable text. See diagram:

- **taJustify:** Aligns the human readable text to both left and right within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren't included when align the human readable text. See diagram:

- **taLeftQuietZone:** Aligns the human readable text to the left within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren included when align the human readable text. See diagram:

- **taCenterQuietZone:** Centers the human readable text horizontally within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren included when align the human readable text. See diagram:

- **taRightQuietZone:** Aligns the human readable text to the right within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren included when align the human readable text. See diagram:

- **taJustifyQuietZone:** Aligns the human readable text to both left and right within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren included when align the human readable text. See diagram:

- **taCustom:** For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, displays the human readable text as UPC/EAN standard format. Forother barcode components, it is the same as using the taJustify. See diagram:

See also the "TextAlignment" property.

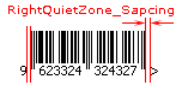
- **TextFont:** TFont; Current value of the TextFont property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the value of the TextFont field of the BarcodeTextDefine parameter). It specifies the font of the human readable text. The color value that's specified by SpaceColor property will be used as the background color (for DrawTo - Syntax 1 and Print - Syntax 1 methods, it's the value of the SpaceColor parameter). See also the "TextFont" property.

- **ExtraFontSize:** Integer; Current value of the ExtraFontSize property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the value of the ExtraFontSize field of the BarcodeTextDefine parameter).

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, and TBarcode1D_UPCE1 barcode components, if the human readable text is represented, and the value of TextAlignment field is equal to taCustom, the parameter specifies the font size of the left quiet zone mark and the right quiet zone mark. Otherwise it will be ignored. The font name, style, and color of the left quiet zone mark and the right quiet zone mark are specified by the TextFont property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the value of the TextFont field of the BarcodeTextDefine parameter). The color value that's specified by the SpaceColor property will be used as the background color (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the value of the SpaceColor parameter). See also the "ExtraFontSize" property. See diagram:

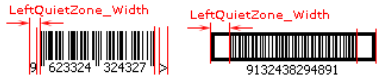
- **LeftQuietZone_Spacing:** Integer; For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components. if the human readable text is represented, and the TextAlignment property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the TextAlignment filed of the BarcodeTextDefine parameter) is set to taCustom (for the TBarcode1D_EAN8 barcode component, the ShowQuietZoneMark property is set to true too), it is the horizontal spacing between the left quiet zone mark and the first bar of barcode symbol, in pixels (Draw, DrawTo) or dots (Print). Otherwise it's zero. See also the LeftQuietZoneSpacing property. See diagram:

- **RightQuietZone_Spacing:** Integer; For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components. if the human readable text is represented, and the TextAlignment property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the TextAlignment field of the BarcodeTextDefine parameter) is set to taCustom (for TBarcode1D_EAN2, TBarcode1D_EAN5, and TBarcode1D_EAN8 barcode components, the ShowQuietZoneMark property is set to true too), it is the horizontal spacing between the last bar of barcode symbol and the right quiet zone mark, in pixels (Draw, DrawTo) or dots (Print). Otherwise it's zero. See also the RightQuietZoneSpacing property. See diagram:



- **LeftQuietZone_Width:** Integer; For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components. if the human readable text is represented, and the TextAlignment property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the TextAlignment field of the BarcodeTextDefine parameter) is set to taCustom (for the TBarcode1D_EAN8 barcode component, the ShowQuietZoneMark property is set to true too), it is the total width of left quiet zone mark and left quiet zone spacing, in pixels (Draw, DrawTo) or dots (Print). For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components. it's the total width of the bearer bar (BearerWidth) and left spacing (LeftSpacing). Otherwise it's zero.

  See diagram:



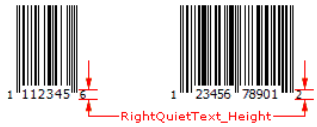
- **RightQuietZone_Width:** Integer; For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components. if the human readable text is represented, and the TextAlignment property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the TextAlignment field of the BarcodeTextDefine parameter) is set to taCustom (for TBarcode1D_EAN2, TBarcode1D_EAN5, and TBarcode1D_EAN8 barcode components, the ShowQuietZoneMark property is set to true too), it is the total width of right quiet zone mark and right quiet

zone spacing, in pixels (Draw, DrawTo) or dots (Print). For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components. it's the total width of right bearer bar (BearerWidth) and right spacing (RightSpacing). Otherwise it's zero.

See diagram:

- **LeftQuietText_Height:** Integer; For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, and TBarcode1D_UPCE1 barcode components. if the human readable text is represented, and the TextAlignment property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the TextAlignment field of the BarcodeTextDefine parameter) is set to taCustom, it is the height of the left quiet zone mark, in pixels (Draw, DrawTo) or dots (Print). Otherwise it's zero. See diagram:

- **RightQuietText_Height:** Integer; For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, and TBarcode1D_UPCE1, barcode components. if the human readable text is represented, and the TextAlignment property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the TextAlignment field of the BarcodeTextDefine parameter) is set to taCustom, it is the height of the right quiet zone mark, in pixels (Draw, DrawTo) or dots (Print). Otherwise it's zero. See diagram:



- **Symbol_Width:** Integer; It's the total width of the barcode symbol, in pixels (Draw, DrawTo) or dots (Print). The human readable text will not be consulted.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the TextAlignment field of the BarcodeTextDefine parameter) is set to taCustom, the LeftQuietZone_Width and the RightQuietZone_Width (width of left and right quiet zone marks and LeftQuietZone_Spacing and RightQuietZone_Spacing) are included.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the LeftQuietZone_Width and the RightQuietZone_Width (width of width of the left and right bearer bars, left spacing, and right spacing) are included too.

  See diagram:

- **Symbol_Height:** Integer; It's the total height of the barcode symbol, in pixels (Draw, DrawTo) or dots (Print). If the human readable text is represented, the height of the human readable text and its vertical spacing (TextVSpacing property) are included.

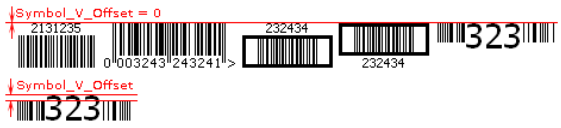
  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars (BearerWidth) are included too.

  If the human readable text is represented, and it exceeds the barcode symbol in vertical direction, the excess isn't included.

  See also the Height property.

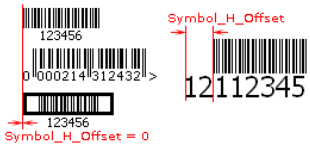
  See diagram:

- **Symbol_V_Offset:** Integer; It's the vertical offset from the upper-left corner of entire barcode (if the human readable text is represented, and it exceeds the barcode symbol in vertical direction, the excess is included) to the upper-left of the barcode symbol (if the human readable text is represented, and it exceeds the barcode symbol in vertical direction, the excess isn't included), in in pixels (Draw, DrawTo) or dots (Print). In general, it is zero. When the human readable text is represented, if the TextPosition property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the TextPosition field of the BarcodeTextDefine parameter) is set to the tpBottomIn or tpBottomOut, and the human readable text exceeds the barcode symbol in vertical direction, the Symbol_V_Offset is greater than zero. See diagram:



- **Symbol_H_Offset:** Integer; It's the horizontal offset from the upper-left corner of entire barcode (if the human readable text is represented, and it exceeds the barcode symbol in horizontal direction, the excess is included) to the upper-left of the barcode symbol (if the human readable text is represented, and it exceeds the barcode symbol in horizontal

direction, the excess isn't included), in in pixels (Draw, DrawTo) or dots (Print). In general, it is zero. When the human readable text is represented, if the TextAlignment property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the TextAlignment field of the BarcodeTextDefine parameter) is set to the tpCenter, tpRight, tpCenterQuietZone, or tpRightQuietZone, and the human readable text exceeds the barcode symbol in horizontal direction, the Symbol_H_Offset is greater than zero. See diagram:



- **Total_Left:** Integer; It's the x-coordinate of the upper-left corner of entire barcode symbol before the barcode symbol is rotated, in pixels (Draw, DrawTo) or dots (Print). If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the TextAlignment field of the BarcodeTextDefine parameter) is set to taCustom, LeftQuietZone_Width and RightQuietZone_Width (width of left and right quiet zone marks and LeftQuietZone_Spacing and RightQuietZone_Spacing) are included in the barcode symbol.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, LeftQuietZone_Width and RightQuietZone_Width (width of the left and right bearer bars, left spacing, and right spacing) are included in the barcode symbol too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal direction, the excess is included in the barcode symbol too.

  See also the "LeftMargin" property.

  See diagram:

- **Total_Top:** Integer; It's the y-coordinate of the upper-left corner of entire barcode symbol before the barcode symbol is rotated, in pixels (Draw, DrawTo) or dots (Print). If the human readable text is represented, the height of the human readable text and its vertical spacing (TextVSpacing property) are included in the barcode symbol.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars (BearerWidth) are included in the barcode symbol too.

  If the human readable text is represented, and it exceeds the barcode symbol in vertical direction, the excess is included in the barcode symbol too.

  See also the "TopMargin" property.

  See diagram:

- **Total_Width:** Integer; It's the width of entire barcode symbol before the barcode symbol is rotated, in pixels (Draw, DrawTo) or dots (Print). If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property (for DrawTo - Syntax 2, DrawTo - Syntax 3, Print - Syntax 2, and Print - Syntax 3 methods, it's the TextAlignment field of the BarcodeTextDefine parameter) is set to taCustom, LeftQuietZone_Width and RightQuietZone_Width (width of left and right quiet zone marks and LeftQuietZone_Spacing and RightQuietZone_Spacing) are included.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, LeftQuietZone_Width and RightQuietZone_Width (width of left and right bearer bars, left spacing, and right spacing) are included too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal direction, the excess is included too.

  See also the "BarcodeWidth" property.

  See diagram:

- **Total_Height:** Integer; It's the height of entire barcode symbol before the barcode symbol is rotated, in pixels (Draw, DrawTo) or dots (Print). If the human readable text is represented, the height of the human readable text and its vertical spacing (TextVSpacing property) are included.

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars (BearerWidth) are included too.

If the human readable text is represented, and it exceeds the barcode symbol in vertical direction, the excess is included too.

See also the "BarcodeWidth" property.

See diagram:

4.3 TBarcodeTextDefine

It contains fields to specify the parameters (e.g. position, alignment , etc.) for the human readable text. It is defined in the pCore1D unit.

The record is used in the following methods:

- DrawTo (Syntax 2)
- DrawToSize (Syntax 2)
- Print (Syntax 2)
- PrintSize (Syntax 2)
- DrawTo (Syntax 3) (only for Delphi/C++ Builder 2009 or later)
- DrawToSize (Syntax 3) (only for Delphi/C++ Builder 2009 or later)
- Print (Syntax 3) (only for Delphi/C++ Builder 2009 or later)
- PrintSize (Syntax 3) (only for Delphi/C++ Builder 2009 or later)

For DrawTo (Syntax 2), DrawTo (Syntax 3), Print (Syntax 2), and Print (Syntax 3) methods, it specifies how to present the human readable text.

For DrawToSize (Syntax 2), DrawToSize (Syntax 3), PrintSize (Syntax 2), and PrintSize (Syntax 3) methods, it's used to calculate the size of the human readable text and the width of quiet zone marks (only for TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components).

For the DrawToSize (Syntax 2) and DrawToSize (Syntax 3) methods, if the Canvas parameter isn't provided or its value is nil, this record will be ignored.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TDisplayText = (dtNone, dtBarcode, dtFullEncoded);
  { Defined in the pCore1D unit }
  TTextPosition = (tpTopIn, tpTopOut, tpBottomIn, tpBottomOut);
  { Defined in the pCore1D unit }
  TTextAlignment = (taLeft, taCenter, taRight, taJustify, taLeftQuietZone, taCenterQuietZone, taRightQuietZone,
    taJustifyQuietZone, taCustom);
  { Defined in the pCore1D unit }
  TBarcodeTextDefine = record
    DisplayText: TDisplayText;
    TextPosition: TTextPosition;
    TextAlignment: TTextAlignment;
    TextFont: TFont;
    ExtraFontSize: Integer;
  end;
```

Fields:

- **DisplayText:** TDisplayText; Specifies whether to display the human readable text and what will display as the human readable text.

  This field can be one of these values (defined in the pCore1D unit):

  - **dtNone:** Don't display the human readable text.
  - **dtBarcode:** Display the barcode text that is specified in the Barcode parameter.
  - **dtFullEncoded:** Display the barcode text that is specified in the Barcode parameter, and the check digit that's automatically appended to the barcode symbol.

  See also the "DisplayText" property.

- **TextPosition:** TTextPosition; Specifies the position of the human readable text (specifies the vertical alignment of the human readable text within the barcode symbol).

  For TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_UPCA, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the TextAlignment field is set to taCustom, the field will be ignored.

  For DrawToSize (Syntax 2), DrawToSize (Syntax 3), PrintSize (Syntax 2), and PrintSize (Syntax 3) methods, the field will be ignored.

This field can be one of these values (defined in the pCore1D unit):

- **tpTopIn:** Justifies the human readable text to the top in the barcode symbol, the bars and spaces on both left and right sides of the human readable text will be reserved. See diagram:

  If the TextAlignment field is set to taJustify or taJustifyQuietZone, it is the same as using the tpTopOut.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, TBarcode1D_ITF16, TBarcode1D_PLANET, TBarcode1D_PostNet, TBarcode1D_JapanPost, TBarcode1D_AP4SC, TBarcode1D_KIX4S, TBarcode1D_RM4SCC, TBarcode1D_PharmacodeTwoTrack, TBarcode1D_PostBar, and TBarcode1D_OneCode barcode components, it is the same as using the tpTopOut.

  For TBarcode1D_EAN2 and TBarcode1D_EAN5 barcode components, if the TextAlignment field is set to taCustom, it is the same as using the tpTopOut.

- **tpTopOut:** Justifies the human readable text to the top in the barcode symbol, the bars and spaces on both left and right sides of the human readable text will be erased. See diagram:

- **tpBottomIn:** Justifies the human readable text to the bottom in the barcode symbol, the bars and spaces on both left and right sides of the human readable text will be reserved.

  If the TextAlignment field is set to taJustify or taJustifyQuietZone, it is the same as using the tpBottomOut.

  For TBarcodeD_ITF6, TBarcodeD_ITF14, TBarcodeD_ITF16, TBarcodeD_PLANET, TBarcodeD_PostNet, TBarcodeD_JapanPost, TBarcodeD_AP4SC, TBarcodeD_KIX4S, TBarcodeD_RM4SCC, TBarcodeD_PharmacodeTwoTrack, TBarcodeD_PostBar, and TBarcodeD_OneCode barcode components, it is the same as using the tpBottomOut.

  For TBarcode1D_EAN2 and TBarcode1D_EAN5 barcode components, if the TextAlignment field is set to taCustom, it is the same as using the tpBottomOut. See diagram:

- **tpBottomOut:** Justifies the human readable text to the bottom in the barcode symbol, the bars and spaces on both left and right sides of the human readable text will be erased. See diagram:

See also the "TextPosition" property.

- **TextAlignment:** TTextAlignment; Determines the horizontal alignment of the human readable text within the barcode symbol.

  This field can be one of these values (defined in the pCore1D unit):

  - **taLeft:** Aligns the human readable text to the left within the barcode symbol. See diagram:

    For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), LeftSpacing, and RightSpacing aren't included when align the human readable text. See diagram:

  - **taCenter:** Centers the human readable text horizontally within the barcode symbol. See diagram:

    For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), LeftSpacing, and RightSpacing

aren't included when align the human readable text. See diagram:

- **taRight:** Aligns the human readable text to the right within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), LeftSpacing, and RightSpacing aren't included when align the human readable text. See diagram:

- **taJustify:** Aligns the human readable text to both left and right within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), LeftSpacing, and RightSpacing aren't included when align the human readable text. See diagram:

- **taLeftQuietZone:** Aligns the human readable text to the left within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), LeftSpacing, and RightSpacing are included when align the human readable text. See diagram:

- **taCenterQuietZone:** Centers the human readable text horizontally within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), LeftSpacing, and RightSpacing are included when align the human readable text. See diagram:

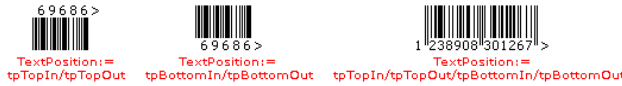
- **taRightQuietZone:** Aligns the human readable text to the right within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), LeftSpacing, and RightSpacing are included when align the human readable text. See diagram:

- **taJustifyQuietZone:** Aligns the human readable text to both left and right within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), LeftSpacing, and RightSpacing are included when align the human readable text. See diagram:

- **taCustom:** For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, displays the human readable text as UPC/EAN standard format. And for TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, the TextPositon field will be ignored. For other barcode components, it is the same as using the taJustify. See diagram:



See also the "TextAlignment" property.

- **TextFont:** TFont; Specifies the font for the human readable text. The color value that's specified by the SpaceColor parameter will be used as the background color.

  If the field is set to nil, the current font of the canvas specified by Canvas parameter will be used.

  See also the "TextFont" property.

- **ExtraFontSize:** Integer; For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, and TBarcode1D_UPCE1 barcode components, if the human readable text is displayed, and the TextAlignment field is set to taCustom, the field specifies the font size for left and right quiet zone marks. Otherwise it will be ignored. The font name, style, and color of the left and right quiet zone marks are specified by the TextFont field. The color value that's specified by the SpaceColor parameter will be used as the background color. See diagram:

If the field value is less than or equal to zero, or greater than the font size that's specified by the TextFont field, the font size that's specified by the TextFont field will be used.

See also the "ExtraFontSize" property.

4.4 TDBBarcode1D

TDBBarcode1D provides an interface between a TDataSource component and a barcode component. It connects the barcode component to a dataset. It's defined in the pDBBarcode1D unit.

Use TDBBarcode1D to provide a conduit between a dataset and a barcode component that enable use the data underlying the dataset to represent the barcode symbol.

See also the "How to use the barcode components with a database".

Properties:

- DataField
- DataSource
- Barcode1D
- BindProperty (*)
- ReadOnly
- Field

(*): The BindProperty property is available only for the Delphi/C++ Builder 2009 or later.

Chapter 5. FAQs

5.1 How to download the full version

Please download the installation package using the download link that's sent from us after you purchase the components package, within the validity period.

If the download link expired, please visit the "Manage your licenses" page to request a new download link.

Please open the page then enter your order ID, license user name or license e-mailaddress to locate your order, then click on the oder ID to display it, choose a license and click on the "**Request a new download link**", the new download link will be sent to this license e-mail address automatically. Please use the new download link whthin the validity period.

5.2 I forgot my license key

If you forgot the license key, please visit the "Manage your licenses" page to retrieve it. Please open the page then enter your order ID, license user name or license e-mail address to locate your order, then click on the oder ID to display it, choose a license and click on the "**Retrieve the license key**", the license key will be sent to the license e-mail address automatically.

Chapter 6. Information

6.1 Support

This help is designed to be used on-screen. Many troubleshooting questions can be answered by this help. If your question is not covered in the help document, please read the next section.

Technical Support

If you have a specific technical issue, if you would like to send general comments about the product, please reads the following:

- Please read the "Frequently Asked Question" in our web site to see whether your question is already answered.
- You may email the technical support issues to support@han-soft.com. The registered users will get priority, so include your order number in the email to ensure a prompt response.

In order to provide more accurate service, please provides the following information:

- Whether the problem can be reproduced? How is it reproduced?
- What development system do you use?
- Which version of 1D Barcode VCL components do you use?

Feedback

If you have any comments or concerns about this product please send them to feedback@han-soft.com. Your feedback is very important for us because it helps to us to make our software better and more efficient. Many of its features have been heavily influenced by comments from users. So if you have a grand idea for a new feature, or a better way of doing something, please drop us a note.

6.2 Purchase

The 1D Barcode VCL Components is a Shareware product. If you find it useful, please purchase the full version ofthis product. After your purchase you will receive an e-mail with a download link of full version for downloading.

Why should I purchase

After your purchase you can continue to use the 1D Barcode VCL Components and entitles you to the following benefits:

- The trial version is fully functional, though limited, and adds a watermark that appears in the final 1D Barcode symbol. So purchase will allow you to use 1D Barcode VCL Components without limitations.
- You are entitled to free upgrades during 1 year since the date of your purchase. This includes both major and minor upgrades in appropriate time period. It means that during one year you can download and install the latest versions of the software from our site.
- You will be entitled to a 50% discount for all future major upgrades if you purchased the software more than one year ago, and allow you to have free upgrades for another year.
- You will be always entitled to free minor upgrades for the version that is purchased.
- You will also have priority in technical support.
- Purchasing the product may also entitle you to discounts on new software releases from Han-soft.
- Finally, by purchasing the software, you provide us with the resources and incentive to support the software with updates and to develop additional quality products in the future.

How to purchase

You can purchase the 1D Barcode VCL Components by using following steps:

1. Please open the "Purchase" web page of our web site http://www.han-soft.com.
2. In the "Purchase" web page, choose the license type and open the order form web page. Fill in the information on the web page, and choose from the available ordering options that best suits your needs. We accept credit card orders, PayPal orders, orders by phone and fax, checks, purchase orders, and wire transfers.
3. Once the purchase is completed, the download link of full version and the license key will be sent instantaneously to your email address. If you do not receive your download link and license key within a few minutes, please check your SPAM filter inbox.

Important:

When completing the order form, please double-check that your email address is correct. If it is not, you will be unable to receive the download link and the license key.

6.3 Contact us

Contact us if you have any questions, comments or suggestions:

Web site

- Main site: http://www.han-soft.com
- Mirror: http://www.han-soft.biz

E-mail

- General information: information@han-soft.com
- Technical support: support@han-soft.com
- Sales: sales@han-soft.com
- Service: service@han-soft.com
- Feedback: feedback@han-soft.com
- Web information: webmaster@han-soft.com

6.4 End-User License Agreement

This End-User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and the Han-soft, which includes computer software and may include associated media, printed materials, and "online" or electronic documentation ("SOFTWARE PRODUCT").

By installing, copying, or otherwise using the SOFTWARE PRODUCT, you agree to be bound by the terms of this EULA. If you do not agree to the terms of this EULA, do not install or use the SOFTWARE PRODUCT; you may, however, return it to your place of purchase for a full refund.

SOFTWARE PRODUCT LICENSE

The SOFTWARE PRODUCT is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The SOFTWARE PRODUCT is licensed, not sold.

1. **GRANT OF LICENSE.**

   This EULA grants you the following rights:

   - **Installation and Use.**

     You may install and use an unlimited number of copies of the TRIAL SOFTWARE PRODUCT.

   - **Reproduction and Distribution.**

     You may reproduce and distribute an unlimited number of copies of the TRIAL SOFTWARE PRODUCT; provided that each copy shall be a true and complete copy, including all copyright and trademark notices, and shall be accompanied by a copy of this EULA. Copies of the TRIAL SOFTWARE PRODUCT may be distributed as a standalone product or included with your own product.

   - **Purchase.**

     You may use the registered SOFTWARE PRODUCT on that number of computers for which you have purchased a separate license as indicated on the invoice or sales receipt. If the SOFTWARE PRODUCT is installed on a network server or other storage device, you must purchase a license for each separate computer on which the SOFTWARE PRODUCT is used. A license for the SOFTWARE PRODUCT may not be shared by alternating use of the SOFTWARE PRODUCT between different computers. The primary user of a computer for which a license has been purchased may make and use one copy of the SOFTWARE PRODUCT on his or her portable computer. You may also make one copy of the SOFTWARE PRODUCT for back-up or archival purposes. Otherwise, you may not copy the SOFTWARE PRODUCT in whole or in part. You may not transfer your rights under this license.

2. **DESCRIPTION OF OTHER RIGHTS AND LIMITATIONS.**

   - **Limitations on Components and Source Code.**

     You may distribute the runtime packages with your end-user applications. You must not distribute the library in such a way that it allows direct programmatic access to the SOFTWARE PRODUCT, or such that it competes with the SOFTWARE PRODUCT in any way - this library is intended for inclusion in end user products. You may not publish or otherwise expose the licensed source code outside of your organization whatsoever. You may modify the source code as required for your own software products but the code may only be used to produce compiled software. Software built incorporating our licensed source code must not allow programmatic access to the software components or in any way compete with our software products.

   - **Limitations on Reverse Engineering, Decompilation, and Disassembly.**

     You may not reverse engineer, decompile, or disassemble the SOFTWARE PRODUCT, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation.

   - **Separation of Components.**

     The SOFTWARE PRODUCT is licensed as a single product. Its components parts may not be separated for use on more than one computer.

   - **Software Transfer.**

     You may permanently transfer all of your rights under this EULA, provided the recipient agrees to the terms of this EULA.

   - **Termination.**

     Without prejudice to any other rights, the Author of this Software may terminate this EULA if you fail to comply with the terms and conditions of this EULA. In such event, you must destroy all copies of the SOFTWARE PRODUCT and all of its components parts.

- **Distribution.**

  The SOFTWARE PRODUCT may not be sold or be included in a product or package which intends to receive benefits through the inclusion of the SOFTWARE PRODUCT. The registered SOFTWARE PRODUCT may not be included in any free or non-profit packages or products.

3. **COPYRIGHT.**

   All title and copyrights in and to the SOFTWARE PRODUCT (including but not limited to any images, photographs, animations, video, audio, music, text, and "applets" incorporated into the SOFTWARE PRODUCT), the accompanying printed materials, and any copies of the SOFTWARE PRODUCT are owned by the Author of this Software. The SOFTWARE PRODUCT is protected by copyright laws and international treaty provisions. Therefore, you must treat the SOFTWARE PRODUCT like any other copyrighted material except that you may install the SOFTWARE PRODUCT on a single computer provided you keep the original solely for backup or archival purposes.

MISCELLANEOUS

Should you have any questions concerning this EULA, or if you desire to contact the author of this Software for any reason, please contact him at the email address mentioned at the bottom of this EULA.

LIMITED WARRANTY

1. **NO WARRANTIES.**

   The Author of this Software expressly disclaims any warranty for the SOFTWARE PRODUCT. The SOFTWARE PRODUCT and any related documentation is provided "as is" without warranty of any kind, either express or implied, including, without limitation, the implied warranties or merchantability, fitness for a particular purpose, or non-infringement. The entire risk arising out of use or performance of the SOFTWARE PRODUCT remains with you.

2. **NO LIABILITY FOR DAMAGES.**

   In no event shall the author of this Software be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising out of the use of or inability to use this product, even if the Author of this Software has been advised of the possibility of such damages. Because some states/jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply to you.

CONTACT US

Please sent email to the sales@han-soft.com email address.

Annex A. Properties

A.1 TBarcode1D

A.1.1 AutoCheckDigit

Specifies whether the check digit should be automatically appended to the barcode symbol.

Syntax:

```
property AutoCheckDigit: Boolean;
```

Description:

Specifies whether the check digit should be automatically appended to the barcode symbol. A check digit is a form of redundancy check used for error detection. It consists of a single character computed from the other character in the barcode text.

A.1.2 Barcode

Specifies the barcode text.

Syntax:

```
property Barcode: string;
```

Description:

Specifies the barcode text to encode it into the barcode symbol. If the property AutoCheckDigit is set to true, the check digit doesn't need to be entered in the here, otherwise the check digit can be specified by you in here. The OnChange event will occur when the property value is changed. The OnInvalidChar event will occur if there is any invalid character in the Barcode property value, and the OnInvalidLength event will occur if the length of the Barcode property value is invalid.

The property is of type string. For Delphi/C++ Builder 2007 or early, it is of type AnsiString. For Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString.

For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, if you use them in the Delphi/C++ Builder 2007 or early, the property is an AnsiSting in ANSI encoding. If you want to use other encoding format (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or specify the converted string in the property. Also, you can use the property to encode a block of binary (bytes) data; If you use them in the Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString. By default, the unicode string will be converted to an ANSI encoding string, then be encoded into the barcode symbol. If you want to use other encoding format (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or specify the converted string in the Data property. If you want to encode block of binary (bytes) data, please use the Data property, it is of type AnsiString. Note, the "\" character is used as a escape prefix, so if you want to encode the "\" character, please use the "\\" instead of it.

For the TBarcode1D_Code32 component, First "A" character does not need to be entered in the property. Also, for the TBarcode1D_PZN component, First "PZN" characters do not need to be entered in the property.

A.1.3 BarcodeHeight

Specifies the distance between the top and bottom of a barcode symbol in pixels.

Syntax:

```
property BarcodeHeight: Integer;
```

Description:

Specifies the distance between the top and bottom of a barcode symbol in pixels. If the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of the top and bottom bearer bars (BearerWidth) are included too. See diagram:

If the human readable text is displayed, and it exceeds the barcode symbol in vertical direction, the excess isn't included. See diagram:

The property is set using the following formula:

- When the property Stretch is set to false:

  The BarcodeHeight property will be ignored, the optimization width will be used instead, it's calculated based on the Height property value.

  You can get the height value by using the Size method.

- When the property Stretch is set to true:
  - If the property value is equal to zero:

    When the Orientation property is set to "boLeftRight" or "boRightLeft", the TopMargin property value will be subtracted from the height of the TImage, TQRImage, or TQRGzImage control that's specified by Image property, then the result will be used as the final barcode height, the barcode symbol will be reduced/stretched to fit this final height value.

    When the Orientation property is set to "boTopBottom" or "boBottomTop", the LeftMargin property value will be subtracted from the width of the TImage, TQRImage, or TQRGzImage control that's specified by the Image property, then the result will be used as the final barcode height, the barcode symbol will be reduced/stretched to fit this final height value.

  - If the property value is greater than zero:

    The barcode symbol will be reduced/stretched to fit this height value.

  - If the property value is less than zero:

    When the Orientation property is set to "boLeftRight" or "boRightLeft", the TopMargin property value and the absolute value of the negative height will be subtracted from the height of the TImage, TQRImage, or TQRGzImage control that's specified by the Image property, then the result will be used as the final barcode height, the barcode symbol will be reduced/stretched to fit this final height value (it specifies the bottom margin of the barcode symbol, -1 denotes the bottom margin is 1, -2 denotes the bottom margin is 2, ...).

    When the Orientation property is set to "boTopBottom" or "boBottomTop", the LeftMargin property value and the absolute value of the negative height will be subtracted from the width of the TImage, TQRImage, or TQRGzImage control that's specified by the Image property, then the result will be used as the final barcode height, the barcode symbol will be reduced/stretched to fit this final height value (it specifies the right margin of the barcode symbol, -1 denotes the right margin is 1, -2 denotes the right margin is 2, ...).

A.1.4 BarcodeWidth

Specifies the distance between the beginning and end of a barcode symbol in pixels.

Syntax:

```
property BarcodeWidth: Integer;
```

Description:

Specifies the distance between the beginning and end of a barcode symbol in pixels. See diagram:

For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the TextAlignment property is set to "taCustom", the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of the left and right bearer bars (BearerWidth), the left spacing, and the right spacing are included. See diagram:

If the human readable text is displayed, and it exceeds the barcode symbol in horizontal direction, the excess is included. See diagram:

The property is set using the following formula:

- When the property Stretch is set to false:

  The BarcodeWidth property will be ignored, the optimization width will be used instead, it's calculated based on the Module property value.

  You can get the optimization width value by using the Size method.

- When the property Stretch is set to true:

  - If the property value is equal to zero:

    When the Orientation property is set to "boLeftRight" or "boRightLeft", the LeftMargin property value will be subtracted from the width of the TImage, TQRImage, or TQRGzImage control that's specified by Image property, then the result will be used as the final barcode width, the barcode symbol will be reduced/stretched to fit this final width value.

    When the Orientation property is set to "boTopBottom" or "boBottomTop", the TopMargin property value will be subtracted from the height of the TImage, TQRImage, or TQRGzImage control that's specified by the Image property, then the result will be used as the final barcode width, the barcode symbol will be reduced/stretched to fit this final width value.

  - If the property value is greater than zero:

    The barcode symbol will be reduced/stretched to fit this width value.

  - If the property value is less than zero:

    When the Orientation property is set to "boLeftRight" or "boRightLeft", the LeftMargin property value and the absolute value of the negative width will be subtracted from the width of the TImage, TQRImage, or TQRGzImage control that's specified by the Image property, then the result will be used as the final barcode width, the barcode symbol will be reduced/stretched to fit this final width value (it specifies the right margin of the barcode symbol, -1 denotes the right margin is 1, -2 denotes the right margin is 2, ...).

    When the Orientation property is set to "boTopBottom" or "boBottomTop", the TopMargin property value and the absolute value of the negative width will be subtracted from the height of the TImage, TQRImage, or TQRGzImage control that's specified by the Image property, then the result will be used as the final barcode width, the barcode symbol will be reduced/stretched to fit this final width value (it specifies the bottom margin of the barcode symbol, -1 denotes the bottom margin is 1, -2 denotes the bottom margin is 2, ...).

A.1.5 BarColor

Specifies the color for all bars in the barcode symbol.

Syntax:

```
property BarColor: TColor;
```

Description:

Specifies the color for all bars in the barcode symbol. By default, it's clBlack.

A.1.6 BarColors

(TBarcode1D_PharmacodeOneTrack, TBarcode1D_PharmacodeTwoTrack)

Specifies the colors of every bars for TBarcode1D_PharmacodeOneTrack and TBarcode1D_PharmacodeTwoTrack components.

Syntax:

```
type
  { Defined in the pPharmacodeOneTrack unit }
  TPharmacodeColors = array[0..15] of TColor;
property BarColors: TPharmacodeColors;
```

Description:

The PharmacodeOneTrack and PharmacodeTwoTrack barcode symbols can be printed in multiple colors as a check to ensure that the remainder of the packaging is correctly printed. The property is a colors array (defined in the pPharmacodeOneTrack unit). It specifies the colors of every bars. The BarColors[0] specifies the right-most bar for barcode symbol. See diagram:

A.1.7 BearerBars

(TBarcode1D_ITF6, TBarcode1D_ITF14, TBarcode1D_ITF16)

Specifies how to represent the the bearer bars of the ITF-6, ITF-14, or ITF-16 barcode symbol.

Syntax:

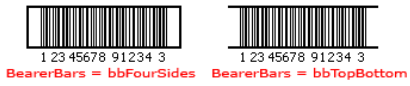
```
type
  { Defined in the pITF6 and pITF14 units }
  TBearerBars = (bbFourSides, bbTopBottom);
property BearerBars: TBearerBars;
```

Description:

The ITF-6, ITF-14, or ITF-16 barcode often employ a bearer bar around the code to protect the symbol from excessive plate squash. The property specifies how to represent the the bearer bar. This property is useful only for TBarcode1D_ITF6, the TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components. It can be one of these values (defined in the pITF6 and pITF14 units):

- **bbFourSides:** A bearer box is represented.
- **bbTopBottom:** Only the top and bottom bearer bars are represented.

See diagram:



A.1.8 BearerWidth

(TBarcode1D_ITF6, TBarcode1D_ITF14, TBarcode1D_ITF16)

Specifies the bearer bar width in modules.

Syntax:

```
property BearerWidth: Integer;
```

Description:

The ITF-6, ITF-14, or ITF-16 barcode often employ a bearer bar around the code to protect the symbol from excessive plate squash. The property specifies the bearer bar width in pixels. This property is useful only for TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components. See diagram:

A.1.9 Bidirectional

(TBarcode1D_Plessey)

Indicates the TBarcode1D_Plessey component to create the bidirection plessey barcode symbol.

Syntax:

```
property Bidirection: Boolean;
```

Description:

Set the property to true in order to create the Plessey Bidirectional barcode symbol. And set the property to false to create the Plessey Unidirectional barcode symbol. This property is useful only for the TBarcode1D_Plessey barcode component.

A.1.10 Channel

(TBarcode1D_Channel)

Specifies the channel for Channel code barcode symbol.

Syntax:

```
type
  { Defined in the pChannel unit }
  TChannel = Byte;
```

```
property Channel: TChannel;
```

Description:

The channel is determined to be one more than the number of digits given in the barcode text in the Channel code barcode symbolpgy. There are 6 channels from 3 to 8, the barcode can hold numbers from any of the following ranges:

- **Channel 3**: 0-26
- **Channel 4**: 0-292
- **Channel 5**: 0-3493
- **Channel 6**: 0-44072
- **Channel 7**: 0-576688
- **Channel 8**: 0-7742862

The property specifies the channel for Channel code barcode symbol. It can be set to a channel number from 3 to 8, or the 0. If you set it to 0, the channel will be selected automatically base on the barcode text. You can use the CurrentChannel property to get the factual channel number.

Note, If the barcode text is out of the number range determined by the channel, an OnInvalidLength or OnInvalidDataLength (only for Delphi/C++ Builder 2009 or later) event will occur.

A.1.11 CheckLength

(TBarcode1D_EAN128)

Specifies the length of the barcode piece which will be used to calculate the check digit in an EAN-128 barcode symbol.

Syntax:

```
property CheckLength: Integer;
```

Description:

In the EAN-128 barcode symbology, an extra check digit is required for some AI fields, you can set the AutoCheckDigit property to true to automatically calculate the check digit. The CheckLength property specifies the length of the barcode piece which will be used to calculate the check digit in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property.

Note, if an EAN-128 barcode contains more than one type of information. CheckStart and CheckLength properties will only work for last data field.

A.1.12 CheckMethod

(TBarcode1D_MSI)

Controls how to calculate the check digit in the TBarcode1D_MSI barcode component.

Syntax:

```
type
  { Defined in the pMSI unit }
  TCheckMethod = (cmMod10, cmMod11, cmMod1010, cmMod1110);
property CheckMethod: TCheckMethod;
```

Description:

The MSI barcode symbology uses one of four possible schemes for calculating a check digit. The property specifies which scheme will be used for calculating a check digit if the AutoCheckDigit property is set to true. It can be one of these values (defined in the pMIS unit):

- **cmMod10:** Using the Mod 10 check digit algorithm.
- **cmMod11:** Using the Mod 11 check digit algorithm.
- **cmMod1010:** Simply calculate the Mod 10 check digit the first time and then calculate it again with the previous result and append the result of the second Mod 10 calculation to the string to be encoded.
- **cmMod1110:** Same as Mod 1010 but the first calculation should be a Mod 11 Check digit.

A.1.13 CheckStart

(TBarcode1D_EAN128)

Specifies the start position of the barcode piece which will be used to calculate the check digit in an EAN-128 barcode symbol.

Syntax:

```
property CheckStart: Integer;
```

Description:

In the EAN-128 barcode symbology, an extra check digit is required for some AI fields, you can set the AutoCheckDigit property to true to automatically calculate the check digit. The CheckStart property specifies the start index of the barcode piece which will be used to calculate the check digit in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property.

Note, if an EAN-128 barcode contains more than one type of information. CheckStart and CheckLength properties will only work for last data field. For the CheckStart property, the index of first AI's first character is 1.

A.1.14 CurrentChannel

(TBarcode1D_Channel)

Contains the factual channel number of a Channel code barcode symbol.

Syntax:

```
type
  { Defined in the pChannel unit }
  TChannel = Byte;
property CurrentChannel: TChannel;
```

Description:

Read the property to retrieve the factual channel number of a Channel code barcode symbol. It can be one of values from 3 to 8, denotations the factual channel of a Channel code barcode symbol.

If the Channel property is set to 0, the factual channel of Channel code symbol will be selected automatically, depending on the barcode text. You can read this property to get the factual channel. Otherwise, the value of this property is equal to the value of Channel property.

The property is read only.

A.1.15 CurrentSubSet

(TBarcode1D_Code128, TBarcode1D_EAN128)

Contains the factual initial characters subset of a Code 128 symbol or a EAN-128 symbol.

Syntax:

```
type
  { Defined in the pCode128 unit }
  TCode128SubSet = (cssSubAuto, cssSubSetA, cssSubSetB, cssSubSetC);
property CurrentSubSet: TCode128SubSet;
```

Description:

Read the property to retrieve the factual initial characters subset of a Code 128 symbol or a EAN-128 symbol. It can be one of these values: cssSubSetA, cssSubSetB, and cssSubSetC, corresponding to the initial characters subset A, B and C. They are defined in the pCode128 unit

If the InitialSubSet property is set to cssSubAuto, the initial characters subset will be selected automatically, depending on the barcode text in order to minimize the symbol size. It can be one of these value: cssSubSetA, cssSubSetB, or cssSubSetC. You can read this property to get the factual initial characters subset.

If the InitialSubSet property isn't set to cssSubAuto, the value of this property is equal to the value of InitialSubSet property.

The property is read only.

See also the InitialSubSet property.

A.1.16 Data

Specifies the barcode text in the AnsiString format.

Syntax:

```
property Data: AnsiString;
```

Description:

If you use the Delphi/C++ Builder 2009 or later, you can use the property to specify a barcode text inAnsiString format, and encode it into the barcode symbol. If the property AutoCheckDigit is set to true, the check digit doesn't need to be entered in the here, otherwise the check digit can be specified by you in here. The OnChange event will occur when the property value is changed. The OnInvalidDataChar event will occur if there is any invalid character in the Data property value, and the OnInvalidDataLength event will occur if the length of the Data property value is invalid.

For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, you can use the property to encode a block of binary (bytes) data under Delphi/C++ Builder 2009 or later. If

you want to encode a block of binary (bytes) data into a barcode symbol under Delphi/C++ Builder 2007 or early, please specify it in the Barcode property.

If you use the TBarcode1D_Code128 and the TBarcode1D_EAN128 components in the Delphi/C++ Builder 2009 or later, the Barcode is in fact an UnicodeString instead of AnsiString. By default, the unicode string will be converted to an ANSI encoding string, then be encoded into the barcode symbol. If you want to use other encoding format (for example the UTF-8, UTF-16), please specify the converted string in the Data property or convert it in the OnEncode event. If you use them in the Delphi/C++ Builder 2007 or early, the Barcode property is an AnsiSting in ANSI encoding. If you want to use other encoding format (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or specify the converted string in the Barcode property.

For the TBarcode1D_OneCode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

For the TBarcode1D_FIM component, it is single character that denotes the FIM type, form "A" to "E", the "E" character dnotes an empty barcode symbol. An OnInvalidDataChar event will occur if you set the property to other character, and an OnInvalidDataLength event will occur if you set the property to 2 or more characters, or set it to empty.

For the TBarcode1D_Patch component, it is single character that denotes the Ptach type, and it can be set to "0", "1", "2", "3", "4", "6", and "T", the "0" character dnotes an empty barcode symbol. An OnInvalidDataChar event will occur if you set the property to other character, and an OnInvalidDataLength event will occur if you set the property to 2 or more characters, or set it to empty.

For the TBarcode1D_Code32 component, First "A" character does not need to be entered in the property. Also, for the TBarcode1D_PZN component, First "PZN" characters do not need to be entered in the property.

Note:

The property is available only for the Delphi/C++ Builder 2009 or later.

For the TBarcode1D_Code128 and TBarcode1D_EAN128 components, the "\" character is used as a escape prefix, so if you want to encode the "\" character, please use the "\\" instead of it.

A.1.17 DisplayText

Specifies whether to display the human readable text and what will display as the human readable text.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TDisplayText = (dtNone, dtBarcode, dtFullEncoded);
property DisplayText: TDisplayText;
```

Description:

Specifies whether to display the human readable text and what will display as the human readable text. This property can be one of these values (defined in the pCore1D unit):

- **dtNone:** Don't display the human readable text.
- **dtBarcode:** Display the barcode text that is specified by the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property.
- **dtFullEncoded:** Display the barcode text that is specified by the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, and the check digit that's automatically appended to the barcode symbol.

Note:

For the TBarcode1D_Code128 and TBarcode1D_EAB128 components, you can encode a block of binary (bytes) data into the barcode symbol. In this case, you can use the OnDecodeText event to decode the text from the block of binary (bytes) data in order to output it as the barcode text into the barcode symbol.

A.1.18 EncodeMode

(TBarcode1D_Code128, TBarcode1D_EAN128)

Determines whether to automatically switch the character subset, and automatically encode the extended ASCII characters, base the barcode text in a TBarcode1D_Code128 or TBarcode1D_EAN128 component, in order to minimize the symbol size.

Syntax:

```
type
  { Defined in the pCode128 unit }
  TCode128EncodeMode = (cemAuto, cemManual);
property EncodeMode: TCode128EncodeMode;
```

Description:

The Code 128 and EAN-128 barcode symbologies can encode all 128 characters of ASCII character sets. This is done by switching between all 3 character subsets of Code 128:

- **SubSet A:** Includes characters with ASCII values from 00 to 95 (i.e. all of the standard upper case alphanumeric characters together with the control characters inclusive), and function characters.
- **SubSet B:** Includes characters with ASCII values from 32 to 127 (i.e. all of the standard upper case alphanumeric characters together with the lower case alphabetic characters inclusive), and function characters.
- **SubSet C:** includes the set of 100 digit pairs from 00 to 99 inclusive, as well as seven special characters. This allows numeric data to be encoded, two data digits per symbol

character, at effectively twice the density of standard data.

Note, only an even number of digits can be encoded if using the character subset C.

Characters with ASCII values 128 to 255 in accordance with ISO 8859-1:1998 may also be encoded. This is done by using the **"FNC 4"** symbol together with character subsets A, B and C.

The property determines whether to automatically switch the character subset, and automatically encode the extended ASCII characters, base the barcode text of a TBarcode1D_Code128 or TBarcode1D_EAN128 component, in order to minimize the symbol size. It can be one of these values (defined in the pCod128 unit):

- **cemAuto:** The character subset will be switched automatically base on the barcode text, and the extended ASCII characters will be encoded by automatically insert the **"FNC4"** function symbol, in order to minimize the symbol size.
- **cemManual:** You need to manually switch the character subset, and manually insert the **"FNC4"** function symbol, in order to encode the all 256 characters of standard and extended ASCII character set, by using following function symbols:
  - **CODE A**: Switch to character subset A. Please use the escape sequence **"\a"** to insert the symbol.
  - **CODE B**: Switch to character subset B. Please use the escape sequence **"\b"** to insert the symbol.
  - **CODE C**: Switch to character subset C. Please use the escape sequence **"\c"** to insert the symbol.
  - **SHIFT**: Change the character subset from A to B or B to A for the single character following the **"SHIFT"** escape sequence. Please use the escape sequence **"\s"** to place the symbol to barcode text.
  - **FNC4**: If a single **"FNC 4"** character is used, indicates the following data character in the symbol is a extended ASCII character. A **"SHIFT"** character may follow the **"FNC 4"** character if it is necessary to change character subset for the following data character. Subsequent data characters revert to the standard ASCII character set. If two consecutive **"FNC4"** characters are used, all following data characters are extended ASCII characters until two further consecutive **"FNC4"** characters are encountered or the end of the symbol is reached. If during this sequence of extended encodation a single **"FNC4"** character is encountered it is used to revert to standard ASCII encodation for the next data character only. **"SHIFT"** and character subset characters shall have their normal effect during such a sequence. Please use the escape sequence **"\4"** to place the symbol to barcode text.

Note, you can manually switch the character subset by using the **"CODE A"**, **"CODE B"**, **"CODE C"**, and **"SHIFT"** function symbols even if the EncodeMode property is set to **cemAuto**. Also, you can manual insert the **"FNC4"** symbol to encode the extended ASCII characters even if the EncodeMode property is set to **cemAuto**.

A.1.19 ExtraChar

(TBarcode1D_Telepen)

Use the property to automatically insert the extra characters to the Telepen symbols.

Syntax:

```
type
  { Defined in the pTelepen unit }
  TTelepenExtraChar = (tecNone, tecESC, tecSISO);
property ExtraChar: TTelepenExtraChar;
```

Description:

Some Telepen symbologies require the first character (after start code) be an ASCII Shift In character, and the last character (before stop code) to be a Shift Out character. The ASCII ESC character is required on some Telepen Numeric systems as the first character after to the start code. The property controls the TBarcode1D_Telepen component to insert these characters automatically. This property is useful only for the TBarcode1D_Telepen component. It can be one of these values (defined in the pTelepen unit):

- **tecNone:** Don't insert any extra characters.
- **tecESC:** Automatically insert the ASCII ESC character after the start code.
- **tecSISO:** Automatically insert the Shift In character after the start code, and the Shift Out character before the stop code.

A.1.20 ExtraFontSize

(TBarcode1D_UPCA, TBarcode1D_UPCE, etc.)

Specifies the font size of the left quiet zone mark and the right quiet zone mark.

Syntax:

```
property ExtraFontSize: Integer;
```

Description:

For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, and TBarcode1D_UPCE1 barcode components, if the human readable text is displayed, and the TextAlignment property is set to taCustom, the property specifies the font size of the left quiet zone mark and the right quiet zone mark. Otherwise it will be ignored. See diagram:

The font name, style, and color of the left quiet zone mark and the right quiet zone mark are specified by the TextFont property. The color value that's specified by the SpaceColor property will be used as the background color.

It defaults to 0, meaning that the font size that's specified by the TextFont property. If the property value is set to less than 0, or greater than the font size that's specified by the TextFont property, the font size that's specified by the TextFont property will be used too.

See also the "TextFont" property.

A.1.21 FIMType

(TBarcode1D_FIM)

For TBarcode1D_FIM barcode component, the property determines which FIM patterns to use.

Syntax:

```
type
  { Defined in the pFIM unit }
  TFIMType = string;
property FIMType: TFIMType;
```

Description:

For FIM barcode symbology, there are four FIM patterns ("A", "B", "C" and "D") that can be used. The property determines which FIM to use as follows:

- **A:** FIM A is used for Courtesy Reply Mail (CRM) and Metered Reply Mail (MRM) with a preprinted POSTNET barcode.
- **B:** FIM B is used for business reply mail (BRM) without a preprinted ZIP + 4 barcode.
- **C:** FIM C is used for business reply mail (BRM) with a preprinted ZIP + 4 barcode.
- **D:** FIM D is used only with information based indicia (IBI) postage.
- **E:** Represent an empty FIM barcode symbol.

Note: Only one of upper case alphabet characters A, B, C, D, E is allowed, an OnInvalidChar event will occur if you set the property to other character, and an OnInvalidLength event will occur if you set the property to 2 or more characters, or set it to empty.

A.1.22 FullEncoded

Contains the barcode text and the check digit that's automatically appended to the barcode symbol. The start code and the stop code aren't included.

Syntax:

```
property FullEncoded: string;
```

Description:

Read FullEncoded property to retrieve the barcode text and the check digit that's automatically appended to the barcode symbol. The start code and the stop code aren't included.

For the TBarcode1D_OneCode component, it includes all the tracking and routing, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

For the TBarcode1D_Channel component, it includes the barcode text with leading zero padding. The length of the property value is from 2 to 7, corresponding to the cannnel 3 to 8.

The property is read only.

A.1.23 Guards

(TBarcode1D_Telepen)

Specifies which start code and the stop code will be used in the Telepen barcode symbol.

Syntax:

```
type
  { Defined in the pTelepen unit }
  TTelepenGuards = (tgAuto, tgASCII, tgCustom, tgNoStop);
```

```
property Guards: TTelepenGuards;
```

Description:

The Telepen symbology provides several encode mode, each having its own pair of start code and stop code. But some Telepen system don't use the start code and the stop code to distinguish between the each mode, and always use the ASCII mode's start code and stop code, the interpretation of the data as ASCII or numeric is set only by the configuration of the reading device. The property specifies which start code and the stop code will be used. It can be one of these values (defined in the pTelepen unit):

- **tgAuto:** Automatically select the start code and stop code base on the value of Mode property, each encode mode having its own pair of start code and stop code.
- **tgASCII:** Always use the ASCII mode's start code and stop code even if the Mode property is set to tmNumeric.
- **tgCustom:** Use the special start code and stop code.
- **tgNoStop:** Use the ASCII mode's start code, but don't use a stop code.

A.1.24 Height

Specifies the distance between the top and bottom of a barcode symbol in modules.

Syntax:

```
property Height: Integer;
```

Description:

Specifies the distance between the top and bottom of a barcode symbol in modules. If the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of the top and bottom bearer bars (BearerWidth) are included too. See diagram:

If the human readable text is displayed, and it exceeds the barcode symbol in vertical direction, the excess isn't included. See diagram:

A.1.25 HideCheckDigitsText

(TBarcode1D_PostBar)

Specifies whether to hide the check digits from the human readable text.

Syntax:

```
property HideCheckDigitsText: Boolean;
```

Description:

For the PostBar barcode symbology, the property specifies whether to hide the check digits from the human readable text. If it's set to true, the check digits will be hidden from the human readable text even if the DisplayText property is set to "dtFullEncoded". This property is useful only for the TBarcode1D_PostBar barcode component.

A.1.26 Image

Specifies a TImage, TQRImage, or TQRGzImage control to represent the barcode symbol.

Syntax:

```
property Image: TControl;
```

Description:

Specifies a TImage control to represent the barcode symbol. The barcode picture will not be saved into the DFM file in design time. If you use the component with QuickReport, you may specifiey a TQRImage or TQRGzImage control for the property.

You can link single TImage, TQRImage, or TQRGzImage control to multiple TBarcode1D components in order to display multiple barcode symbols in the TImage, TQRImage, or TQRGzImage control (using the LeftMargin and TopMargin properties to specify the position for every barcode symbol).

A.1.27 InitialSubSet

(TBarcode1D_Code128, TBarcode1D_EAN128)

Specifies an initial characters subset for a TBarcode1D_Code128 component or a TBarcode1D_EAN128 component.

Syntax:

```
type
  { Defined in the pCode128 unit }
  TCode128SubSet = (cssSubAuto, cssSubSetA, cssSubSetB, cssSubSetC);
property InitialSubSet: TCode128SubSet;
```

Description:

The Code 128 and EAN-128 barcode symbologies can encode all 128 characters of ASCII character sets. This is done by switching between all 3 character subsets of Code 128:

- **SubSet A:** Includes characters with ASCII values from 00 to 95 (i.e. all of the standard upper case alphanumeric characters together with the control characters inclusive), and function characters.
- **SubSet B:** Includes characters with ASCII values from 32 to 127 (i.e. all of the standard upper case alphanumeric characters together with the lower case alphabetic characters inclusive), and function characters.
- **SubSet C:** includes the set of 100 digit pairs from 00 to 99 inclusive, as well as seven special characters. This allows numeric data to be encoded, two data digits per symbol character, at effectively twice the density of standard data.

  Note, only an even number of digits can be encoded if using the character subset C.

Use the InitialSubSet property to specify a initial characters subset for aTBarcode1D_Code128 or TBarcode1D_EAN128 component. When the EncodeMode property is set to **cemAuto**, the characters subset will be automatically switched if a character is encountered that cannot be encoded by current characters subset. You need to manually insert the **\a**, **\b**, **\c**, or **\s** for switching the character subset if the EncodeMode property is set to **cemManual**.

The property can be one of these values (defined in the pCode128 unit):

- **cssSubAuto:** The initial characters subset will be selected depending on the barcode text in order to minimize the symbol size. You can always use the CurrentSubSet property to get the factual initial characters subset.
- **cssSubSetA:** Using the characters subset A as the initial characters subset.
- **cssSubSetB:** Using the characters subset B as the initial characters subset.
- **cssSubSetC:** Using the characters subset C as the initial characters subset.

You can always use the CurrentSubSet property to get the factual initial characters subset.

A.1.28 InterGap

(TBarcode1D_PostBar, TBarcode1D_AP4SC, TBarcode1D_ITF14, etc.)

Specifies whether to insert an inter-cipher gap of one module between characters.

Syntax:

```
property InterGap: Boolean;
```

Description:

For TBarcode1D_Code39, TBarcode1D_Code39Ext, TBarcode1D_UPU, TBarcode1D_PZN, and TBarcode1D_Code32 barcode components, the property specifies whether to insert an inter-cipher gap of one module between characters. If the property is set to true, the inter-cipher gap is used. If the property is set to false there is no inter-cipher gap.

A.1.29 LeftMargin

Specifies the left margin of the barcode symbol in pixels.

Syntax:

```
property LeftMargin: Integer;
```

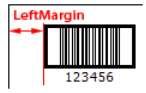
Description:

Specifies the margin between the leftmost side of the barcode symbol and the left side of the TImage, TQRImage, or TQRGzImage control that is specified by the Image property, in pixels. See diagram:

If the human readable text is displayed, it's included in the barcode symbol. See diagram:

For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment property is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too. See diagram:



If the human readable text is displayed and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too. See diagram:

It is set using the following formula:

- The Orientation property is set to "boLeftRight":

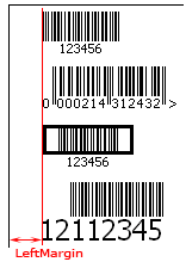
  It is the margin between the beginning of the barcode symbol and the left side of the TImage, TQRImage, or TQRGzImage control.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the TextAlignment property is set to "taCustom", it's the margin between the left quiet zone mark and the left side of the TImage, TQRImage, or TQRGzImage control (The ShowQuietZoneMark property of the TBarcode1D_EAN2, TBarcode1D_EAN5, or TBarcode1D_EAN8 barcode component is set to true).

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, it's the margin between the left bearer bar and the left side of the TImage, TQRImage, or TQRGzImage control.

  When the human readable text is displayed, and the TextAlignment property is set to "taRight", "taRightQuietZone", "taCenter" or "taCenterQuietZone", if the human readable text exceeds the beginning of the barcode symbol, it's the margin between the beginning of the human readable text and the left side of the TImage, TQRImage, or TQRGzImage control.

  See diagram:



- The Orientation property is set to "boRightLeft":

  It is the margin between the end of the barcode symbol and the left side of the TImage, TQRImage, or TQRGzImage control.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the TextAlignment property is set to "taCustom", it's the margin between the right quiet zone mark and the left side of the TImage, TQRImage, or TQRGzImage control (TheShowQuietZoneMark property of the TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, or TBarcode1D_EAN13 barcode component is set to true).

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, it's the margin between the right bearer bar and the left side of the TImage, TQRImage,

or TQRGzImage control.

When the human readable text is displayed, and the TextAlignment property is set to "taLeft", "taLeftQuietZone", "taCenter" or "taCenterQuietZone", if the human readable text exceeds the end of the barcode symbol, it's the margin between the end of the human readable text and the left side of the TImage, TQRImage, or TQRGzImage control.

See diagram:

- The Orientation property is set to "boTopBottom":

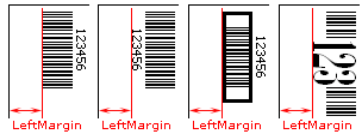
  It is the margin between the bottom of the barcode symbol and the left side of the TImage, TQRImage, or TQRGzImage control.

  If the human readable text is displayed and the TextPosition property is set to the "tpBottomIn" or "tpBottomOut", It's the margin between the bottom of the human readable text and the left side of the TImage, TQRImage, or TQRGzImage control.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, if the human readable text isn't displayed or the TextPosition property is set to the "tpTopIn" or "tpTopOut", it's the margin between the bottom bearer bar and the left side of the TImage, TQRImage, or TQRGzImage control.

  When the human readable text is displayed, and the TextPosition property is set to the "tpTopIn" or "tpTopOut", if the human readable text exceeds the barcode symbol in height, it's the margin between the bottom of the human readable text and the left side of the TImage, TQRImage, or TQRGzImage control.

  See diagram:



- The Orientation property is set to "boBottomTop":

  It is the margin between the top of the barcode symbol and the left side of the TImage, TQRImage, or TQRGzImage control.

  If the human readable text is displayed and the TextPosition property is set to the "tpTopIn" or "tpTopOut", It's the margin between the top of the human readable text and the left side of the TImage, TQRImage, or TQRGzImage control.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, if the human readable text isn't displayed or the TextPosition property is set to the "tpBottomIn" or "tpBottomOut", it's the margin between the top bearer bar and the left side of the TImage, TQRImage, or TQRGzImage control.

  When the human readable text is displayed, and the TextPosition property is set to the "tpBottomIn" or "tpBottomOut", if the human readable text exceeds the barcode symbol in height, it's the margin between the top of the human readable text and the left side of the TImage, TQRImage, or TQRGzImage control.

  See diagram:

A.1.30 LeftQuietZoneSpacing

(TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_EAN13, etc.)

Specifies the horizontal spacing between the left quiet zone mark and the first bar of barcode symbol in modules.

Syntax:

```
property LeftQuietZoneSpacing: Integer;
```

Description:

Specifies the horizontal spacing between the left quiet zone mark and the first bar of barcode symbol in modules. This property is useful only for TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, and the TextAlignment is set to taCustom (For the TBarcode1D_EAN8 barcode component, the ShowQuietZoneMark property is set to true too). See diagram:

A.1.31 LeftSpacing

(TBarcode_ITF6, TBarcode_ITF14, TBarcode_ITF16)

Specifies the horizontal spacing between the left bearer bar and the first bar of barcode symbol in modules.

Syntax:

```
property LeftSpacing: Integer;
```

Description:

Specifies the horizontal spacing between the left bearer bar and the first bar of the ITF-6, ITF-14, or ITF-16 barcode symbol, in modules. This property is useful only for TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components. See diagram:

A.1.32 Link2D

(TBarcode1D_EAN128)

Specifies whether an EAN128 symbol can be used as the linear component of an EAN.UCC composite symbol.

Syntax:

```
type
  { Defined in the pEAN128 unit }
  TLink2D = (ldNone, ldCCA, ldCCB, ldCCC);
property Link2D: TLink2D;
```

Description:

The EAN128 barcode symbol can be used together with a 4-column CC-A, a 4-column CC-B, or a CC-C symbol to create the EAN.UCC composite symbol. The property specifies whether the EAN128 symbol can be used as the linear component of an EAN.UCC composite symbol. It can be one of these values (defined in the pEAN128 unit):

- **ldNone:** The EAN128 symbol will be used as a stand-alone barcode symbol.
- **ldCCA:** A contiguous separator pattern will be represented on top of the EAN128 symbol. It shall be aligned with a "CC-A" 2D component in order to comprising a singleEAN.UCC Composite symbol.
- **ldCCB:** A contiguous separator pattern will be represented on top of the EAN128 symbol. It shall be aligned with a "CC-B" 2D component in order to comprising a singleEAN.UCC Composite symbol.
- **ldCCC:** A contiguous separator pattern will be represented on top of the EAN128 symbol. It shall be aligned with a "CC-C" 2D component in order to comprising a singleEAN.UCC Composite symbol.

Normally the EAN128 symbol, its contiguous separator pattern, and the 2D component are represented at the same time, comprising a single EAN.UCC composite symbol.

If you use the EAN128 component together with the **CC-A**, **CC-B**, or **CC-C** 2D component that is in the **2D Barcode VCL Components**package, the property will be set automatically depending on the 2D component.

A.1.33 Locked

Set the property to true to prevents updating of the barcode component until the property is reset to false.

Syntax:

```
property Locked: Boolean;
```

Description:

Set the property to true before making multiple changes to the barcode component. When all changes are complete, change the property to false so that the changes can be reflected on screen. It prevents excessive redraws and speed processing time when change the barcode component.

Set the property to True is similar to BeginUpdate method of a other Delphi/C++ Builder control such as a ListBox, Memo, TreeList, ListView, and set the property to False is similar to its EndUpdate method.

A.1.34 Mod11Weighting

(TBarcode1D_MSI)

Controls which repeated weighting factor patterns will be used to calculate the Mod 11 check digit in the TBarcode1D_MSI barcode component.

Syntax:

```
type
  { Defined in the pMSI unit }
  TMod11Weighting = (mwIBM, mwNCR);
property Mod11Weighting: TMod11Weighting;
```

Description:

For TBarcode1D_MSI barcode component, the Mod 11 check digit algorithm uses one of two different repeated weighting factor patterns for calculating a check digit. The property specifies which repeated weighting factor patterns will be used to calculate the Mod 11 check digit. It's useful only when the value of AutoCheckDigit property is set to true, and the value of CheckMethod property is set to cmMod11 or cmMod1110. It can be one of these values (defined in the pMSI unit):

- **mwIBM:** The IBM (International Business Machines Corporation) repeating weighting factor pattern will be used.
- **mwNCR:** The NCR (National Cash Register Company) repeating weighting factor pattern will be used.

A.1.35 Mode

(TBarcode1D_Telepen)

Specifies the initial encode mode for the Telepen barcode symbol.

Syntax:

```
type
  { Defined in the pTelepen unit }
  TTelepenMode = (tmFullASCII, tmASCII, tmNumeric);
property Mode: TTelepenMode;
```

Description:

There are three encode modes for the Telepen barcode symbology. The property specifies the initial encode mode. It can be one of these values (defined in the pTelepen unit):

- **tmFullASCII:** Encodes all ASCII characters (ASCII 0 - ASCII 127). It cannot be switched to other encode modes.
- **tmASCII:** Encodes all ASCII characters, a DLE character (ASCII 16) will switch to the Numeric mode.
- **tmNumeric:** Encodes numeric data in double-density mode, a DLE character (ASCII 16) will switch to the ASCII mode.

Note, the mode switching is permitted only once in a Telepen symbol.

A.1.36 Module

Specifies the module width in pixels.

Syntax:

```
property Module: Integer;
```

Description:

Specifies the module width in pixels, it is the width of the smallest bar (or space) in the barcode symbol. See diagram:

A.1.37 NumberCheckDigit

(TBarcode1D_Code11)

Specifies the nubmer of check digits of the TBarcode1D_Code11 barcode component.

Syntax:

```
type
  { Defined in the pCode11 unit }
  TNumberCheckDigit = (ncdAuto, ncdOne, ncdTwo);
property NumberCheckDigit: TNumberCheckDigit;
```

Description:

Specifies the nubmer of check digits of the TBarcode1D_Code11 barcode component, it's useful only when the AutoCheckDigit property is set to true. This property can be one of these values (defined in the pCode11 unit):

- **ncdAuto:** If the length of the data is longer than 10 characters, two check digits are used, named C and K. If the length of the data is 10 characters or fewer, only the first check digit (C) is used.
- **ncdOne:** Only use the first check digit, C.
- **ncdTwo:** Use two check digits, C and K.

A.1.38 NumCustomerInfo

(TBarcode1D_AP4SC)

Specifies which encoding table will be used to encode the Customer Information Field in a TBarcode1D_AP4SC barcode component.

Syntax:

```
property NumCustomerInfo: Boolean;
```

Description:

The property specifies which encoding table will be used to encode the Customer Information Field in a TBarcode1D_AP4SC barcode component.

The Customer Information Field is only available when the first 2 digits (FCC: Format Control Code) of Barcode or Data (only for Delphi/C++ Builder 2009 or later) property are 59, 62, or 44. It includes all characters from 11th character to end of the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, and enables customers to include their own information in the barcode.

If the value of NumCustomerInfo property is set to true, only the digits "0" to "9" can be used in the Customer Information Field. If the value of NumCustomerInfo property is set to false, the uppercase character "A" to "Z", lowercase characters "a" to "z", digits "0" to "9", " " (space) and "#" characters can be used in the Customer Information Field.

The maximum length of the Customer Information field is specified in the following table:

FCC	NumCustomerInfo := True	NumCustomerInfo := False
44	15 digits	10 characters
59	8 digits	5 characters
62	15 digits	10 characters

A.1.39 OddEncode

(TBarcode1D_Telepen)

Controls how to encode an odd number of digits in the Numeric mode of the Telepen barcode symbology.

Syntax:

```
type
  { Defined in the pTelepen unit }
  TTelepenOddEncode = (toeUseNULL, toeToASCII, toeUseZero);
property OddEncode: TTelepenOddEncode;
```

Description:

In the Numeric mode of the Telepen barcode symbology, only an even number of digits can be encoded. The property controls how to encode an odd number of digits. It can be one of these values (defined in the pTelepen unit):

- **toeUseNULL:** It's encoded by using the "1X" to "9X" characterts in the last digit of the numeric mode piece.
- **toeToASCII:** It's encoded automatically inserting a DLE character (ASCII 16) before the last digit of the numeric mode piece to switch to the ASCII mode. Since the mode switching is permitted only once in a Telepen symbol, if current numeric mode is switching from ASCII mode, the "1X" to "9X" characters will be used instead of a switching to ASCII mode.
- **toeUseZero:** It's encoded adding a "0" as the first character of the numeric mode piece.

A.1.40 Orientation

Controls the orientation of the barcode symbol.

Syntax:

```
type
  { Defined in the pBarcode1D unit }
  TBarcodeOrientation = (boLeftRight, boRightLeft, boTopBottom, boBottomTop);
property Orientation: TBarcodeOrientation;
```

Description:

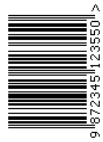
Specifies the direction of the barcode symbol. This property can be one of these values (defined in the pBarcode1D unit):

- **boLeftRight:** Left to right horizontal direction (rotates the barcode symbol 0 degrees counter-clockwise). See diagram:

- **boRightLeft:** Right to left horizontal direction (rotates the barcode symbol 180 degrees counter-clockwise). See diagram:

- **boTopBottom:** Top to bottom vertical direction (rotates the barcode symbol 270 degrees counter-clockwise). See diagram:

- **boBottomTop:** Bottom to top vertical direction (rotates the barcode symbol 90 degrees counter-clockwise). See diagram:



A.1.41 Padding

(TBarcode1D_Code25Interleaved)

Controls how to encode an odd number of digits in the TBarcode1D_Code25Interleaved barcode component.

Syntax:

```
type
  { Defined in the pCode25Int unit }
  TCode25InterleavedPadding = (cipLeft, cipRight);
property Padding: TCode25InterleavedPadding;
```

Description:

For the Code 25 Interleaved barcode symbology, only an even number of digits can be encoded. The property controls how to encode an odd number of digits. This property can be one of these values (defined in the pCode25Int unit):

- **cipLeft:** It's encoded by adding a "0" as first digit.
- **cipRight:** It's encoded using five narrow spaces in the last digit.

A.1.42 PatchType

(TBarcode1D_Patch)

For TBarcode1D_Patch barcode component, the property determines which Patch Code patterns to use.

Syntax:

```
type
  { Defined in the pPatch unit }
  TPatchType = string;
property PatchType: TPatchType;
```

Description:

For Patch Code barcode symbology, there are six patch patterns ("1", "2", "3", "4", "6" and "T") that can be used. The property determines which Patch Code to use as follows:

- **1:** Creates the Patch Type 1. It can be used for by the host for post-scan image control for the i800/i1800 (with image addressing) Series Scanners (they are not used for image addressing).
- **2:** Creates the Patch Type 2. It is used for assigning image level 2 to the current document.
- **3:** Creates the Patch Type 3. It is used for assigning image level 3 to the current document.
- **4:** Create the Patch Type 4 / Toggle Patch. The Patch Type 4 is used by the host for post-scan image control for the i800/i1800 (with image addressing) Series Scanners (they are not used for image addressing). The Toggle Patch is used to switch back and forth from bi-tonal and color/grayscale scanning for the i280, 3590C, i600, i800 and i1800 (without image addressing) Series Scanners. This provides Color on the Fly during capture, with no need for post-scan processing by the host application.
- **6:** Creates the Patch Type 6. It can be used for by the host for post-scan image control for the i800/i1800 (with image addressing) Series Scanners (they are not used for image addressing).
- **T:** Creates the Patch T / Transfer Patch. It can be used to assigns a predefined image level to the next document. The predefined image level is based upon the transfer patch definition which is defined for each application. For example, if the transfer patch definition is image level 2, then use of a transfer patch assigns image level 2 to the next document.

Note: Only one of characters 1, 2, 3, 4, 5, and T is allowed, an OnInvalidChar event will occur if you set the property to other character, and an OnInvalidLength event will occur if you set the property to 2 or more characters, or set it to empty.

A.1.43 Ratio

Specifies the bar(space) width ratio for barcode symbol.

Syntax:

```
property Ratio: Double;
```

Description:

Specifies ratio between a wide bar (or space) and a narrow bar (or space) in the barcode symbol. The normal values are from 2.0 to 3.0. See diagram:

A.1.44 RightQuietZoneSpacing

(TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_EAN13, etc.)

Specifies the horizontal spacing between the last bar of barcode symbol and the right quiet zone mark in modules.

Syntax:

```
property RightQuietZoneSpacing: Integer;
```

Description:

Specifies the horizontal spacing between the last bar of barcode symbol and the right quiet zone mark in modules. This property is useful only for TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, and the TextAlignment property is set to taCustom (For TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, the ShowQuietZoneMark property is set to true too). See diagram:

A.1.45 RightSpacing

(TBarcode1D_ITF6, TBarcode1D_ITF14, TBarcode1D_ITF16)

Specifies the horizontal spacing between the last bar of barcode symbol and the right bearer bar in modules.

Syntax:

```
property RightSpacing: Integer;
```

Description:

Specifies the horizontal spacing between the right bearer bar and the last bar of the ITF-6, ITF-14, or ITF-16 barcode symbol, in modules. This property is useful only for TBarcode1D_ITF6, and the TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components. See diagram:

A.1.46 Routing

(TBarcode1D_OneCode)

Specifies the routing code for the OneCode barcode symbol.

Syntax:

```
property Routing: string;
```

Description:

Specifies the routing code for the TBarcode1D_OneCode barcode component. The routing code shall consist of the delivery point ZIP Code. The Delivery Point ZIP Code shall be assigned by the mailer for routing the mailpiece. The length may be 0, 5, 9, or 11 digits. The allowable encoding ranges shall be no ZIP Code, 00000-99999, 000000000-999999999, and 00000000000-99999999999.

Example:

12345, 123456789, 12345678901.

A.1.47 ShortFinder

(TBarcode1D_Channel)

Specifies whether to use the shortened finder pattern in a Channel code barcode symbol.

Syntax:

```
property ShortFinder: Boolean;
```

Description:

For the TBarcode1D_Channel barcode component, the property specifies whether to use the shortened finder pattern.

A.1.48 ShowGuards

(TBarcode1D_Code39, TBarcode1D_UPU, TBarcode1D_Code32, etc.)

Specifies whether to display the left and right guard characters (start character and stop character).

Syntax:

```
property ShowGuards: Boolean;
```

Description:

For TBarcode1D_Code39, TBarcode1D_Code39Ext, TBarcode1D_UPU, TBarcode1D_PZN, and TBarcode1D_Code32 barcode components, the property specifies whether to display the left and right guard characters (start character and stop character). If the property is set to true, the start and stop characters are shown as an asterisk (*) in the human readable text. If the property is set to false then the start and stop characters are not represented in human readable text. Note, you don't have to specify the start and stop characters to the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property even if they are displayed in the human readable text.

A.1.49 ShowQuietZoneMark

(TBarcode1D_ENA5, TBarcode1D_EAN8, TBarcode1D_EAN13, etc.)

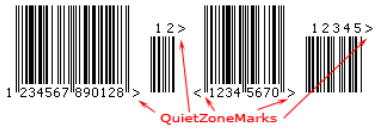
Specifies whether to display the left and right quiet zone marks.

Syntax:

```
property ShowQuietZoneMark: Boolean;
```

Description:

For TBarcode1D_EAN2, TBarcode1D_ENA5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, the property specifies whether to display the left and right quiet zone marks. It's useful only when the DisplayText property is not set to dtNone, and the TextAlignment property is set to taCustom. See diagram:



Note:

For TBarcode1D_EAN2 and TBarcode1D_ENA5 barcode components, the left quiet zone mark does not exist.

For the TBarcode1D_EAN13 barcode component, the left quiet zone mark (number system character) is displayed always even if the property is set to false.

For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, and TBarcode1D_UPCE1 barcode components, the left quiet zone mark (number system character or prefix digit) and right quiet zone mark (symbol check character) are displayed always, so the property does not be provided for these components.

A.1.50 SpaceColor

Specifies the color for all spaces in the barcode symbol.

Syntax:

```
property SpaceColor: TColor;
```

Description:

Specifies the color for all spaces in the barcode symbol. By default, it's clWhite.

If the human readable text is represent, the color will be used as its background color (the foreground color is specified using the TextFont property). For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 components, the left spacing and the right spacing will be represented using the color. For TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_UPCA, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 components, if the human readable text is represented, and the TextAlignment property is set to taCustom, the left and right quiet zones, left quietzone spacing and right quiet zone spacing will be represented using the color. In other words, the property specify the background color for entire barcode symbol.

A.1.51 SplitText

(TBarcode1D_PostBar, TBarcode1D_AP4SC, TBarcode1D_ITF14, etc.)

Determines whether to split the human readable text base on data fields.

Syntax:

```
property SplitText: Boolean;
```

Description:

The property determines whether to split the human readable text base on data fields. Set the property to true to insert space characters between all data fields in the human readable text. This property is useful only for TBarcode1D_PostBar, TBarcode1D_AP4SC, TBarcode1D_ITF14, TBarcode1D_JapanPost, TBarcode1D_UPU, TBarcode1D_Leitcode, and TBarcode1D_Identcode barcode components.

For the TBarcode1D_OneCode component, it isn't a boolean property, see also the SplitText property of TBarcode1D_OneCode component.

A.1.52 SplitText

(TBarcode1D_OneCode)

Determines how to split the human readable text of OneCode barcode symbology, base on data fields.

Syntax:

```
type
  { Defined in the pOneCode unit }
  TOneCodeSplitTextMode = (stmNone, stmMID6SN9, stmMID9SN6);
property SplitText: TOneCodeSplitTextMode ;
```

Description:

The property determines how to split the human readable text of OneCode barcode symbology, base on data fields. It can be one of these values (defined in the pOneCode unit):

- **stmNone:** Don't insert space characters between all data fields in the human readable text.
- **stmMID6SN9:** Insert space characters between all data fields in the human readable text, the mailer or customer identifier field is 6 digit number, and the serial or sequence number field is 9 digit number.
- **stmMID9SN6:** Insert space characters between all data fields in the human readable text, the mailer or customer identifier field is 9 digit number, and the serial or sequence number field is 6 digit number.

A.1.53 StartCode

(TBarcode1D_Codabar)

Specifies the start code for the Codabar barcode symbol to mark the beginning of the barcode.

Syntax:

```
type
  { Defined in the pCodabar unit }
  {$IFDEF DELPHI_7_AND_UPPER}
  TFrameChars = 'A'..'D';
  {$ELSE}
  TFrameChars = string;
  {$ENDIF}
property StartCode: TFrameChars;
```

Description:

Specifies the start code for the Codabar barcode symbol to mark the beginning of the barcode. It's valid only for the TBarcode1D_Codabar component. This property can be one of alphabet characters A, B, C, D, and in some specifications, they are named E, N, asterisk, and T. They are defined in the pCodabar unit.

Note: Only one of upper case alphabet characters A, B, C, D is allowed. For Delphi 7 or upper, C++ Builder 2006 or upper, the ERangeError exception will occur if the value of the property is not from "A" to "D" (the exception will be suppress if you close the "Range checking" in project options). For Delphi 6 or lower, C++ Builder 6 or lower, operation will be ignored if you set the property to other character or characters string.

A.1.54 StopCode

(TBarcode1D_Codabar)

Specifies the stop code for the Codabar barcode symbol to mark the end of the barcode.

Syntax:

```
type
  { Defined in the pCodabar unit }
  {$IFDEF DELPHI_7_AND_UPPER}
  TFrameChars = 'A'..'D';
  {$ELSE}
  TFrameChars = string;
  {$ENDIF}
property StopCode: TFrameChars;
```

Description:

Specifies the stop code for the Codabar barcode symbol to mark the end of the barcode. It's valid only for the TBarcode1D_Codabar component. This property can be one of alphabet characters A, B, C, D, and in some specifications, they are named E, N, asterisk, and T. They are defined in the pCodabar unit.

Note: Only one of upper case alphabet characters A, B, C, D is allowed. For Delphi 7 or upper, C++ Builder 2006 or upper, the ERangeError exception will occur if the value of the property is not from "A" to "D" (the exception will be suppress if you close the "Range checking" in project options). For Delphi 6 or lower, C++ Builder 6 or lower, operation will be ignored if you set the property to other character or characters string.

A.1.55 Stretch

Specifies whether to reduce/stretch the barcode symbol to specified size.

Syntax:

```
property Stretch: Boolean;
```

Description:

The property specifies whether to reduce/stretch the barcode symbol to fit the size specified by BarcodeWidth and BarcodeHeight properties.

- If the property is set to false, the barcode symbol will not be reduced/stretched. The values of BarcodeWidth and BarcodeHeight properties will be ignored.

  The barcode symbol width will be calculated based on the Module property value.

  The barcode symbol height will be calculated based on the Height property value.

  You can get the width and height by using the Size method.

- If the property is set to true, the barcode symbol will be reduced/stretched to fit the size specified by BarcodeWidth and BarcodeHeight properties.

  The BarcodeWidth property specifies the width of barcode symbol. If the property value is less than or equal to zero, it specifies the right margin of the barcode symbol (the Orientation property is set to "boLeftRight" or "boRightLeft"), or the bottom margin of the barcode symbol (the Orientation property is set to "boTopBottom" or "boBottomTop"). See also the "BarcodeWidth" property.

  The BarcodeHeight property specifies the height of barcode symbol. If the property value is less than or equal to zero, it specifies the bottom margin of the barcode symbol (the Orientation property is set to "boLeftRight" or "boRightLeft"), or the right margin of the barcode symbol (the Orientation property is set to "boTopBottom" or "boBottomTop"). See also the "BarcodeHeight" property.

A.1.56 StretchTextHeight

Specifies whether to reduce/stretch the barcode text when the barcode symbol be reduced/stretched.

Syntax:

```
property StretchTextHeight: Boolean;
```

Description:

The property specifies whether to reduce/stretch the barcode text when the barcode symbol be reduced/stretched to fit the height specified by the BarcodeHeight property. Note, the property is valid only when the Stretch property is set to true, otherwise, the StretchTextHeight property's value will e ignored.

See diagram:

A.1.57 TextAlignment

Determines the horizontal alignment of the human readable text within the barcode symbol.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TTextAlignment = (taLeft, taCenter, taRight, taJustify, taLeftQuietZone, taCenterQuietZone, taRightQuietZone,
    taJustifyQuietZone, taCustom);
property TextAlignment: TTextAlignment;
```

Description:

Determines the horizontal alignment of the human readable text within the barcode symbol. This parameter can be one of these values (defined in the pCore1D unit):

- **taLeft:**

  Aligns the human readable text to the left within the barcode symbol. See diagram:

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren't included when align the human readable text. See diagram:

- **taCenter:**

  Centers the human readable text horizontally within the barcode symbol. See diagram:

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren't included when align the human readable text. See diagram:

- **taRight:**

  Aligns the human readable text to the right within the barcode symbol. See diagram:

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren't included when align the human readable text. See diagram:

- **taJustify:**

  Aligns the human readable text to both left and right within the barcode symbol. See diagram:

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren't included when align the human readable text. See diagram:

- **taLeftQuietZone:**

  Aligns the human readable text to the left within the barcode symbol. See diagram:

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren included when align the human readable text. See diagram:

- **taCenterQuietZone:**

  Centers the human readable text horizontally within the barcode symbol. See diagram:

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren included when align the human readable text. See diagram:

- **taRightQuietZone:**

  Aligns the human readable text to the right within the barcode symbol. See diagram:

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren included when align the human readable text. See diagram:

- **taJustifyQuietZone:**

  Aligns the human readable text to both left and right within the barcode symbol. See diagram:

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the LeftSpacing, and the RightSpacing aren included when align the human readable text. See diagram:

- **taCustom:**

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, displays the human readable text as UPC/EAN standard format. For other barcode components, it is the same as using the taJustify. See diagram:

A.1.58 TextCSpacing

Specifies the intercharacter spacing for the human readable text, in modules.

Syntax:

```
property TextCSpacing: Integer;
```

Description:

It's the amount of extra space, in modules, to be added to each character in the human readable text. This property is useful only when the DisplayText property isn't set to dtNone, and the TextAlignment property is set to taLeft, taCenter, taRight, taLeftQuietZone, taCenterQuietZone, or taRightQuietZone. See diagram:

A.1.59 TextFont

Specifies the font of the human readable text.

Syntax:

```
property TextFont: TFont;
```

Description:

Specifies the font of the human readable text. The color value that's specified by SpaceColor property will be used as the background color.

Note, In the TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, and TBarcode1D_UPCE1 barcode components, if the TextAlignment property is set to taCustom, for the left quiet zone mark and right quiet zone mark, the property specifies the font name, style, color, etc but the font size. The font size of the left quiet zone mark and the right quiet zone mark is specified by the "ExtraFontSize" property. See also the "ExtraFontSize" property.

A.1.60 TextHSpacing

Specifies the horizontal spacing between the barcode symbol and the human readable text in modules.

Syntax:

```
property TextHSpacing: Integer;
```

Description:

Specifies the horizontal spacing between the barcode symbol and the human readable text in modules. This property is useful only when the DisplayText property isn't set to dtNone, the TextPosition property is set to tpTopIn or tpBottomIn, and the TextAlignment property is set totaLeft, taRight, taCenter, or taCustom. See diagram:

A.1.61 TextPosition

Specifies the position of the human readable text (Specifies the vertical alignment of the human readable text within the barcode symbol).

Syntax:

```
type
  { Defined in the pCore1D unit }
  TTextPosition = (tpTopIn, tpTopOut, tpBottomIn, tpBottomOut);
property TextPosition: TTextPosition;
```

Description:

Specifies the position of the human readable text (Specifies the vertical alignment of the human readable text within the barcode symbol). This property can be one of these values (defined in the pCore1D unit):

- **tpTopIn:**

  Justifies the human readable text to the top in the barcode symbol, the bars and spaces on both left and right sides of the human readable text will be reserved. See diagram:

  If the TextAlignment property is set to taJustify or taJustifyQuietZone, it is the same as using the tpTopOut.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, TBarcode1D_ITF16, TBarcode1D_PLANET, TBarcode1D_PostNet, TBarcode1D_AP4SC, TBarcode1D_KIX4S, TBarcode1D_RM4SCC, TBarcode1D_PharmacodeTwoTrack, TBarcode1D_PostBar, and TBarcode1D_OneCode barcode components, it is the same as using the tpTopOut.

  For TBarcode1D_EAN2 and TBarcode1D_EAN5 barcode components, if the TextAlignment property is set to taCustom, it is the same as using the tpTopOut.

- **tpTopOut:**

  Justifies the human readable text to the top in the barcode symbol, the bars and spaces on both left and right sides of the human readable text will be erased. See diagram:

- **tpBottomIn:**

  Justifies the human readable text to the bottom in the barcode symbol, the bars and spaces on both left and right sides of the human readable text will be reserved. See diagram:

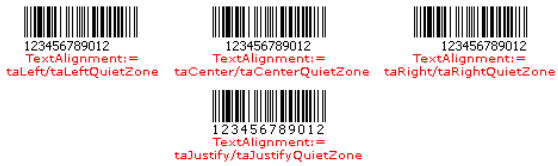
  If the TextAlignment property is set to taJustify or taJustifyQuietZone, it is the same as using the tpBottomOut.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, TBarcode1D_ITF16, TBarcode1D_PLANET, TBarcode1D_PostNet, TBarcode1D_AP4SC, TBarcode1D_KIX4S, TBarcode1D_RM4SCC, TBarcode1D_PharmacodeTwoTrack, TBarcode1D_PostBar, and TBarcode1D_OneCode barcode components, it is the same as using the tpBottomOut.

  For TBarcode1D_EAN2 and TBarcode1D_EAN5 barcode components, if the TextAlignment property is set to taCustom, it is the same as using the tpBottomOut.

- **tpBottomOut:**

  Justifies the human readable text to the bottom in the barcode symbol, the bars and spaces on both left and right sides of the human readable text will be erased. See diagram:



Note:

For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the TextAlignment property is set to taCustom, the value of this property will be ignored.

For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN8, TBarcode1D_EAN13, and TBarcode1D_EAN128 barcode components, if they are used as the **Linear** property's value of the **TBarcode2D_CCA**, **TBarcode2D_CCB**, or **TBarcode2D_CCC** 2D component in the **2D Barcode VCL Components** package to generate the EAN.UCC composite barcode symbol, the property value **tpTopIn** is equal to the **tpBottomIn**, and the **tpTopOut** is equal to the **tpBottomOut**.

A.1.62 TextVSpacing

Specifies the vertical spacing between the barcode symbol and the human readable text in modules.

Syntax:

```
property TextVSpacing: Integer;
```

Description:

Specifies the vertical spacing between the barcode symbol and the human readable text in modules. This property is useful only when the DisplayText property isn't set to dtNone. See diagram:

A.1.63 TopMargin

Specifies the top margin of the barcode symbol in pixels.

Syntax:

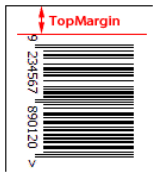
```
property TopMargin: Integer;
```

Description:

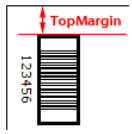
Specifies the margin between the topmost of the barcode symbol and the top side of the TImage, TQRImage, or TQRGzImage control that is specified by the Image property, in pixels. See diagram:

If the human readable text is displayed, it's included in the barcode symbol. See diagram:

For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment property is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too. See diagram:



For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too. See diagram:



If the human readable text is displayed and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too. See diagram:

It is set using the following formula:

- The Orientation property is set to "boLeftRight":

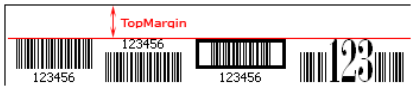
  It is the margin between the top of the barcode symbol and the top side of the TImage, TQRImage, or TQRGzImage control.

  If the human readable text is displayed and the TextPosition property is set to the "tpTopIn" or "tpTopOut", It's the margin between the top of the human readable text and the top side of the TImage, TQRImage, or TQRGzImage control.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, if the human readable text isn't displayed or the TextPosition property is set to the "tpBottomIn" or "tpBottomOut", it's the margin between the top bearer bar and the top side of the TImage, TQRImage, or TQRGzImage control.

  When the human readable text is displayed, and the TextPosition property is set to the "tpBottomIn" or "tpBottomOut", if the human readable text exceeds the barcode symbol in height, it's the margin between the top of the human readable text and the top side of the TImage, TQRImage, or TQRGzImage control.

  See diagram:



- The Orientation property is set to "boRightLeft":

  It is the margin between the bottom of the barcode symbol and the top side of the TImage, TQRImage, or TQRGzImage control.

  If the human readable text is displayed and the TextPosition property is set to the "tpBottomIn" or "tpBottomOut", It's the margin between the bottom of the human readable text and the top side of the TImage, TQRImage, or TQRGzImage control.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, if the human readable text isn't displayed or the TextPosition property is set to the "tpTopIn" or "tpTopOut", it's the margin between the bottom bearer bar and the top side of the TImage, TQRImage, or TQRGzImage control.

  When the human readable text is displayed, and the TextPosition property is set to the "tpTopIn" or "tpTopOut", if the human readable text exceeds the barcode symbol in height, it's the margin between the bottom of the human readable text and the top side of the TImage, TQRImage, or TQRGzImage control.

  See diagram:

- The Orientation property is set to "boTopBottom":

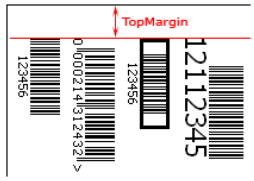
  It is the margin between the beginning of the barcode symbol and the top side of the TImage, TQRImage, or TQRGzImage control.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the TextAlignment property is set to "taCustom", it's the margin between the left quiet zone mark and the top side of the TImage, TQRImage, or TQRGzImage control (The ShowQuietZoneMark property of the TBarcode1D_EAN2, TBarcode1D_EAN5, or TBarcode1D_EAN8 barcode component is set to true).

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, it's the margin between the left bearer bar and the top side of the TImage, TQRImage, or TQRGzImage control.

  When the human readable text is displayed, and the TextAlignment property is set to "taRight", "taRightQuietZone", "taCenter" or "taCenterQuietZone", if the human readable text exceeds the beginning of the barcode symbol, it's the margin between the beginning of the human readable text and the top side of the TImage, TQRImage, or TQRGzImage control.

See diagram:



- The Orientation property is set to "boBottomTop":

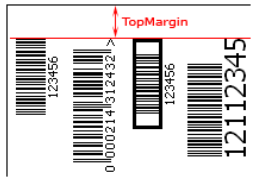
  It is the margin between the end of the barcode symbol and the top side of the TImage, TQRImage, or TQRGzImage control.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the TextAlignment property is set to "taCustom", it's the margin between the right quiet zone mark and the top side of the TImage, TQRImage, or TQRGzImage control (The ShowQuietZoneMark property of the TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, or TBarcode1D_EAN13 barcode component is set to true).

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, it's the margin between the right bearer bar and the top side of the TImage, TQRImage, or TQRGzImage control.

  When the human readable text is displayed, and the TextAlignment property is set to "taLeft", "taLeftQuietZone", "taCenter" or "taCenterQuietZone", if the human readable text exceeds the end of the barcode symbol, it's the margin between the end of the human readable text and the top side of the TImage, TQRImage, or TQRGzImage control.

  See diagram:



A.1.64 Tracking

(TBarcode1D_OneCode)

Specifies the tracking code for OneCode barcode symbol.

Syntax:

```
property Tracking: string;
```

Description:

Specifies the tracking code for the TBarcode1D_OneCode barcode component. It's 20 digits, and it shall consist of the Barcode Identifier, Service Type Identifier, Mailer Identifier, and Serial Number fields in the order specified in the following list:

- **Barcode Identifier:** This shall be assigned by USPS to encode the presort identification that is currently printed in human readable form on the optional endorsement line (OEL) as well as for future USPS use. This shall be two digits, with the second digit in the range of 0-4. The allowable encoding ranges shall be 00-04, 10-14, 20-24, 30-34, 40-44, 50-54, 60-64, 70-74, 80-84, and 90_94. The valid field values are 00, 10, 20, 30, 40, and 50.
- **Service Type Identifier:** This shall be assigned by USPS for any combination of services requested on the mailpiece. The allowable encoding range shall be 000-999. Each 3-digit value shall correspond to a particular mail class with a particular combination of service(s). Each service program, such as OneCode Confirm and OneCode ACS, shall provide the list of Service Type Identifier values. The valid field values when no services are requested are 700, 702, 704, 706, 708, 710, and 712 now.
- **Mailer Identifier:** The shall be assigned by USPS as a unique, 6 or 9 digit number that identifies a business entity. The allowable encoding range for the 6 digit Mailer Identifier shall be 000000-899999, while the allowable encoding range for the 9 digit Mailer Identifier shall be 900000000-999999999.
- **Serial Number:** This shall be assigned by the mailer for uniquely identifying and tracking mailpieces. The allowable encoding range shall be 000000000-999999999 when used with a 6 digit Mailer Identifier and 000000-999999 when used with a 9 digit Mailer Identifier.

For each of the fields in the list, leading or trailing zeros shall be provided to achieve the correct size.

Example:

10702123456789123456, 20704987654321654321.

A.1.65 UKMode

(TBarcode1D_Plessey)

Specifies whether to create the Plessey barcode symbol in UK (United Kingdom) mode.

Syntax:

```
property UKMode: Boolean;
```

Description:

In other parts of the world Plessey may refer to a barcode similar to but different in detail to the code used in the UK. Set the property to Ture to create thePlessey barcode symbol in UK (United Kingdom) mode. This property is useful only for the TBarcode1D_Plessey component.

TDBBarcode1D

A.2.1 Barcode1D

Links a TBarcode1D component to the TDBBarcode1D component in order to represent the barcode symbol from a data field of the current record of a dataset.

Syntax:

```
property Barcode1D: TBarcode1D;
```

Description:

Use the Barcode1D property to specify aTBarcode1D component such as the TBarcode1D_Code39, the TBarcode1D_EAN13, and the TBarcode1D_ITF14, It will represent the barcode symbol from a data field (specified by the DataField property of the TDBBarcode1D component) of the current record of a dataset (specified by the DataSource property of the TDBBarcode1D component) to a TImage, TQRImage, or TQRGzImage control that's specified by the Image property of the TBarcode1D component.

For Delphi/C++ Builder 2007 or early, when aTBarcode1D barcode component is linked to the TDBBarcode1D component, except TBarcode1D_FIM, TBarcode1D_Patch, and TBarcode1D_OneCode components, the data field value will be applied to its Barcode property. If a TBarcode1D_FIM component is linked to the TDBBarcode1D component, The first character of the data field value will be applied to its FIMType property. If a TBarcode1D_Patch component is linked to the TDBBarcode1D component, The first character of the data field value will be applied to its PatchType property. If a TBarcode1D_OneCode component is linked to the TDBBarcode1D component, the data field value will applied to its Tracking and Routing properties, the first 20 characters are the Tracking (it is right padded with zeroes to 20 characters), then come the Routing.

For Delphi/C++ Builder 2009 or later, the value ofBindProperty property will indicate which property the data field value will be applied to. When a TBarcode1D barcode component is linked to the TDBBarcode1D component, except TBarcode1D_FIM, TBarcode1D_Patch, and TBarcode1D_OneCode components, if the BindProperty property is set to bpBarcode, the data field value will be applied to its Barcode property. If the BindProperty property is set to bpData, the data field value will be applied to its Data property. For the TBarcode1D_FIM component, if the BindProperty property is set to bpBarcode, the first character of the data field value will be applied to its FIMType property. If the BindProperty property is set to bpData, the data field value will be applied to its Data property (only first byte will be used). For the TBarcode1D_Patch component, if the BindProperty property is set to bpBarcode, the first character of the data field value will be applied to its PatchType property. If the BindProperty property is set to bpData, the data field value will be applied to its Data property (only first byte will be used). For the TBarcode1D_OneCode component, if the BindProperty property is set to bpBarcode, the data field value will applied to its Tracking and Routing properties, the first 20 characters are the Tracking (it is right padded with zeroes to 20 characters), then come the Routing. If the BindProperty property is set to bpData, the data field value will be applied to its Data property (first 20 bytes are the tracking, it is right padded with zeroes to 20 bytes, then come the routing).

A.2.2 BindProperty

Indicates which property of the TBarcode1D component the data field value will be applied to, in order to represent the barcode symbol from a data field of the current record of a dataset.

Syntax:

```
type
  { Defined in the pDBBarcode1D unit }
  TBindProperty = (bpBarcode, bpData);
property BindProperty: TBindProperty;
```

Description:

The Barcode property of TBarcode1D component is of type string, for Delphi/C++ Builder 2009 or later, it is in fact anUnicodeString instead of AnsiString. In order to encode the barcode text in AnsiString format, we added a Data property, it is of type AnsiString.

When a TBarcode1D barcode component is linked to the TDBBarcode1D component, it will represent the barcode symbol from a data field (specified by the DataField property of the TDBBarcode1D component) of the current record of a dataset (specified by the DataSource property of the TDBBarcode1D component). The BindProperty property indicates which property of the TBarcode1D component the data field value will be applied to.

This property can be one of these values (defined in the pDBBarcode1D unit):

- **bpBarcode**: The data field value will be applied to the Barcode property of the TBarcode1D barcode component (for the TBarcode1D_FIM component, it's the FIMType property; for the TBarcode1D_Patch component, it's the PatchType property; for the TBarcode1D_OneCode component, it's the Tracking and Routing properties, the first 20 characters are the Tracking, it is right padded with zeroes to 20 characters, then come the Routing).
- **bpData**: The data field value will be applied to the Data property of the TBarcode1D barcode component.

Note:

The property is available only for the Delphi/C++ Builder 2009 or later.

A.2.3 DataField

Specifies the field from which the TDBBarcode1D component represents barcode symbol.

Syntax:

```
property DataField: string;
```

Description:

Use DataField to bind the TDBBarcode1D component to a field in the dataset. To fully specify a data field, both the dataset and the field within that dataset must be defined. The DataSource property of the TDBBarcode1D component specifies the dataset which contains the data field. The data field should be specifed as a string, bytes, blob, memo, etc field.

For Delphi/C++ Builder 2007 or early, when a TBarcode1D barcode component is linked to the TDBBarcode1D component, except TBarcode1D_FIM, TBarcode1D_Patch, and TBarcode1D_OneCode components, the data field value will be applied to its Barcode property. If a TBarcode1D_FIM component is linked to the TDBBarcode1D component, The first character of the data field value will be applied to its FIMType property. If a TBarcode1D_Patch component is linked to the TDBBarcode1D component, The first character of the data field value will be applied to its PatchType property. If a TBarcode1D_OneCode component is linked to the TDBBarcode1D component, the data field value will applied to its Tracking and Routing properties, the first 20 characters are the Tracking (it is right padded with zeroes to 20 characters), then come the Routing.

For Delphi/C++ Builder 2009 or later, the value of BindProperty property will indicate which property the data field value will be applied to. When a TBarcode1D barcode component is linked to the TDBBarcode1D component, except TBarcode1D_FIM, TBarcode1D_Patch, and TBarcode1D_OneCode components, if the BindProperty property is set to bpBarcode, the data field value will be applied to its Barcode property. If the BindProperty property is set to bpData, the data field value will be applied to its Data property. For the TBarcode1D_FIM component, if the BindProperty property is set to bpBarcode, the first character of the data field value will be applied to its FIMType property. If the BindProperty property is set to bpData, the data field value will be applied to its Data property (only first byte will be used). For the TBarcode1D_Patch component, if the BindProperty property is set to bpBarcode, the first character of the data field value will be applied to its PatchType property. If the BindProperty property is set to bpData, the data field value will be applied to its Data property (only first byte will be used). For the TBarcode1D_OneCode component, if the BindProperty property is set to bpBarcode, the data field value will applied to its Tracking and Routing properties, the first 20 characters are the Tracking (it is right padded with zeroes to 20 characters), then come the Routing. If the BindProperty property is set to bpData, the data field value will be applied to its Data property (first 20 bytes are the tracking, it is right padded with zeroes to 20 bytes, then come the routing).

You can bind multiple TDBBarcode1D components to one data field in order to represent the data field with multiple barcode symbols.

A.2.4 DataSource

Links the TDBBarcode1D component to the dataset that contains the field it represents.

Syntax:

```
property DataSource: TDataSource;
```

Description:

Use DataSource to specify the data source component through which the data from a dataset component is provided to the TDBBarcode1D component. To fully specify a data field for the TDBBarcode1D component, both the dataset and a field within that dataset must be defined. Use the DataField property to specify the particular field within the dataset.

You can bind multiple TDBBarcode1D components to one data field in order to represent the data field with multiple barcode symbols.

A.2.5 Field

Indicates the TField object whose current value the barcode component represents.

Syntax:

```
property Field: TField;
```

Description:

Use the TField reference provided by the Field property when you want to read or to write the value of the data in the field programmatically.

A.2.6 ReadOnly

Determines whether the user can change the value of the field.

Syntax:

```
property ReadOnly: Boolean;
```

Description:

Use the ReadOnly property to specify whether the TBarcode1D barcode component specified in the Barcode1D property of a TDBBarcode1D component allows the user to change the field value by changing the value of its Barcode property (for the TBarcode1D_FIM component, it's the FIMType property; for the TBarcode1D_Patch component, it's the PatchType property; for the TBarcode1D_OneCode component, it's the Tracking and Routing properties, the first 20 characters are the Tracking, it is right padded with zeroes to 20 characters, then come the Routing), or its Data property (only for Delphi/C++ Builder 2009 or later).

When the ReadOnly property is set to true, the TBarcode1D barcode component can only be used to represent the value of the field on the current record.

When the ReadOnly property is set to false, for the Delphi/C++ Builder 2007 or early, the user can change the field value by changing the Barcode property value of the TBarcode1D barcode component specified in the Barcode1D property of the TDBBarcode1D component (for the TBarcode1D_FIM component, it's the FIMType property; for the TBarcode1D_Patch component, it's the PatchType property; for the TBarcode1D_OneCode component, it's the Tracking and Routing properties, the first 20 characters are the Tracking, it is right padded with zeroes to 20 characters, then come the Routing). For the Delphi/C++ Builder 2009 or later, if the BindProperty is set to bpBarcode, the user can change the field value by changing the Barcode property value of the TBarcode1D barcode component specified in the Barcode1D property of the TDBBarcode1D component (for the TBarcode1D_FIM component, it's the FIMType property; for the TBarcode1D_Patch component, it's the PatchType property; for the TBarcode1D_OneCode component, it's the Tracking and Routing properties, the first 20 characters are the Tracking, it is right padded with zeroes to 20 characters, then come the Routing). If the BindProperty is set to bpData, the user can change the field value by changing the Data property value of the TBarcode1D barcode component specified in the Barcode1D property of the TDBBarcode1D component.

Annex B. Methods

B.1 TBarcode1D

B.1.1 Assign

Copies a barcode component from another barcode component.

Syntax:

```
procedure Assign(Source: TPresistent); override;
```

Description:

If the Source parameter is an object created from a subclass of TBarcode1D component class, and the class is same to current barcode component class, Assign copies all property values and event handles except the Locked and Image properties from the source barcode component to current one. If Source is any other type of object, an EBarcode1DError exception occurs.

Parameters:

- **Source:** TPersistent; Specifies the source object.

B.1.2 Clear

Erases current barcode symbol in the TImage, TQRImage, or TQRGzImage control that's specified by the Image property.

Syntax:

```
function Clear(UseSpaceColor: Boolean = False): Boolean; virtual;
```

Description:

The method erases the barcode symbol without erasing the background around it from the TImage, TQRImage, or TQRGzImage control that's specified by the Image property.

Parameters:

- **UseSpaceColor:** Boolean; Specifies whether the space color that's specified by SpaceColor property is used to erase the barcode symbol. If it's set to false, the current brush color of the TImage, TQRImage, or TQRGzImage control will be used. If the parameter isn't provided, it's default to be false.

Return:

- If the method succeeds, the return value is true.
- If the Image property is not set, the return value is false.

  If the length of Barcode or Data (only for Delphi/C++ Builder 2009 or later) property value is invalid, the return value is false, corresponding to the OnInvalidLength or OnInvalidDataLength (only for Delphi/C++ Builder 2009 or later) event will occur.

  If there is any invalid character in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property value, the return value is false, corresponding to the OnInvalidChar or OnInvalidDataChar (only for Delphi/C++ Builder 2009 or later) event will occur.

Note:

For Delphi 3, the default value of parameters isn't supported, so the UseSpaceColor parameter is required.

B.1.3 CopyToClipboard

Copies a barcode symbol to the system clipboard. There are several different overloading methods, Syntax 1, Syntax 2, and Syntax 3 (only for Delphi/C++Builder 2009 or later):

- **Syntax 1:** Copies the barcode symbol that is specified in the properties of this barcode component to the system clipboard.
- **Syntax 2:** Copies the barcode symbol that is specified in the parameters of this method to the system clipboard. The barcode text is specified in the Barcode parameter. It is of type string. For Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiString. For Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, if you use them in the Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiSting in ANSI encoding scheme. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or specify the converted string in the Barcode paramater. Also, you can use the method to encode a block of binary (bytes) data; If you use them in the Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString

instead of AnsiString. By default, the unicode string will be converted to an ANSI encoding string, then beencoded into the barcode symbol. If you want to use other encoding scheme (for example the UTF-8, UTF-16),please convert it in the OnEncode event, or use the CopyToClipboard (Syntax 3) overloading method and specify the converted string in its Data paramater. If you want to encode block of binary (bytes) data, please use the CopyToClipboard (Syntax 3) overloading method.

- **Syntax 3:** Copies the barcode symbol that is specified in the parameters of this method to the system clipboard. The barcode text is specified in the Data parameter. It is of type AnsiString, so you can specifiy the barcode text in AnsiString format. The method is available only for the Delphi/C++ Builder 2009 or later.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, you can use the method to encode a block of binary (bytes) data under Delphi/C++ Builder 2009 or later. If you want to encode a block of binary (bytes) data into a barcode symbol under Delphi/C++ Builder 2007 or early, please use the CopyToClipboard (Syntax 2) overloading method.

Note:

For Delphi 3, the method overload isn't supported, so the method names of Syntax 1 and Syntax 2 are changed to CopyToClipboard1 and CopyToClipboard2.

B.1.3.1 CopyToClipboard - Syntax 1

Copies a barcode symbol to the system clipboard. The barcode symbol is specified in the properties of this barcode component.

Syntax:

```
function CopyToClipboard(Module: Integer = 0; Height: Integer = 0; Angle: Integer = -1): Integer; overload; virtual;
```

Description:

Copies current barcode symbol that is specified in the properties of this barcode component to the system clipboard.

Parameters:

- **Module:** Integer; Specifies the module width in logical dots or pixels in the horizontal direction, itis the width of the smallest bar (or space) in the barcode symbol. If the parameter isn't provided or it is set to zero, the value of "Module" property will be used.

  See also the "Module" property.

- **Height:** Integer; Specifies the distance between the top and bottom of a barcode symbol in modules. Ifthe human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included.

  If the parameter isn't provided or its value is set to zero, the value of Height property will be used.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of the top and bottom bearer bars (BearerWidth) are included too.

  If the human readable text is displayed, and it exceeds the barcode symbol in vertical direction, the excess isn't included.

  See also the "Height" property.

  Note: If the parameter is less than zero, its absolute value specifies the height in pixels in the vertical direction.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to -1 if theAngle is not provided, and the barcode symbol will be rotated base on the value of the Orientation property:

  - boLeftRight: 0 degrees
  - boRightLeft: 180 degrees
  - boTopBottom: 270 degrees
  - boBottomTop: 90 degrees

  If you want to use the -1 degrees, the 359 degrees can be used instead.

Return:

- If the method succeeds, the return value is zero.
- If the length of the barcode text that is specified byBarcode or Data (only for Delphi/C++ Builder 2009 or later) property is invalid, the return valueis -1. Corresponding to the OnInvalidLength or OnInvalidDataLength (only for Delphi/C++ Builder 2009 or later) event will occur.
- If there is any invalid character in the the barcode text that is specified by Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character. Corresponding to the OnInvalidChar or OnInvalidDataChar (only for Delphi/C++ Builder 2009 or later) event will occur. For the TBarcode1D_OneCode component, if the invalid character in the Tracking property, the index is from 1 to 20 including 1 and 20; if it is in the Routing property, the value starts with 21 (First character of the Routing property).

See diagram:

Note:

For Delphi 3, the method overload and default value of parameter aren't supported, so the method name is changed to CopyToClipboard1, and the parameters Module, Height, Angle are required.

B.1.3.2 CopyToClipboard - Syntax 2

Copies a barcode symbol to the system clipboard. The barcode symbol is specified in the parameters of this method.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TDisplayText = (dtNone, dtBarcode, dtFullEncoded);
  { Defined in the pCore1D unit }
  TTextPosition = (tpTopIn, tpTopOut, tpBottomIn, tpBottomOut);
  { Defined in the pCore1D unit }
  TTextAlignment = (taLeft, taCenter, taRight, taJustify, taLeftQuietZone, taCenterQuietZone, taRightQuietZone,
    taJustifyQuietZone, taCustom);
function CopyToClipboard(Barcode: string; AutoCheckDigit: Boolean; BarColor, SpaceColor: TColor; BarcodeTextDefine:
  TBarcodeTextDefine; Ratio: Double; Module: Integer = 0; Height: Integer = 0; Angle: Integer = 0): Integer; overload;
  virtual;
```

Description:

Copies a barcode symbol that is specified in the parameters of this method to the system clipboard.

Parameters:

- **Barcode:** String; Specifies the barcode text. It is of type string. For Delphi/C++ Builder 2007 orearly, the Barcode parameter is in fact an AnsiString. For Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, if you use them in the Delphi/C++ Builder 2007 orearly, the Barcode parameter is in fact an AnsiSting in ANSI encoding scheme. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in theOnEncode event, or specify the converted string in the Barcode paramater. Also, you can use the method to encode a block of binary (bytes) data; If you use them in the Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString. By default, the unicode string will be converted to an ANSI encoding string, then beencoded into the barcode symbol. If you want to use other encoding scheme (for example the UTF-8, UTF-16),please convert it in the OnEncode event, or use the CopyToClipboard (Syntax 3) overloading method and specify the converted string in its Data paramater. If you want to encode block of binary (bytes) data, please use the CopyToClipboard (Syntax 3) overloading method. Note, the "\" character is used as a escape prefix, so if you want to encode the "\" character, please use the "\\" instead of it.

  For the TBarcode1D_OneCode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

  For the TBarcode1D_FIM component, it is single character that denotes the FIM type, form "A" to "E", the "E" character dnotes an empty barcode symbol.

  For the TBarcode1D_Patch component, it is single character that denotes the Ptach type, and it can be set to "0", "1", "2", "3", "4", "6", and "T", the "0" character dnotes an empty barcode symbol.

  For the TBarcode1D_Code32 component, First "A" character does not need to be entered in the parameter. Also, for the TBarcode1D_PZN component, First "PZN"characters do not need to be entered in the parameter.

  See also the "Barcode" property.

- **AutoCheckDigit:** Boolean; Specifies whether the check digit should be automatically appended to the barcode symbol.

  See also the "AutoCheckDigit" property.

- **BarColor:** TColor; Specifies the color for all bars in the barcode symbol.

  See also the "BarColor" property.

- **SpaceColor:** TColor; Specifies the color for all spaces in the barcode symbol.

  If the human readable text is represent, the color will be used as its background color (the foreground color is specified using the TextFont field of the BarcodeTextDefine parameter). For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 components, the left spacing and the right spacing will be represented using the color. For TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_UPCA, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones, left quietzone spacing and right quiet zone spacing will be represented using the color. In other words, the parameter specify the background color for entire barcode symbol.

  See also the "SpaceColor" property.

- **BarcodeTextDefine:** TBarcodeTextDefine; Specifies whether to display the human readable text and how todisplay the human readable text. The record is defined in the pBarcode1D unit.

  See also the TBarcodeTextDefine record.

- **Ratio:** Double; Specifies ratio between a wide bar (or space) and a narrow bar (or space) in the barcode symbol. The normal values are from 2.0 to 3.0. If the parameter is less than or equal to zero, the method fails, and the return value is -2.

  See also the "Ratio" property.

- **Module:** Integer; Specifies the module width in logical dots or pixels in the horizontal direction, it is the width of the smallest bar (or space) in the barcode symbol. If the parameter isn't provided or it is set to zero, the value of "Module" property will be used.

See also the "Module" property.

- **Height:** Integer; Specifies the distance between the top and bottom of a barcode symbol in modules. Ifthe human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included.

  If the parameter isn't provided or its value is set to zero, the value of Height property will be used.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of the top and bottom bearer bars (BearerWidth) are included too.

  If the human readable text is displayed, and it exceeds the barcode symbol in vertical direction, the excess isn't included.

  See also the "Height" property.

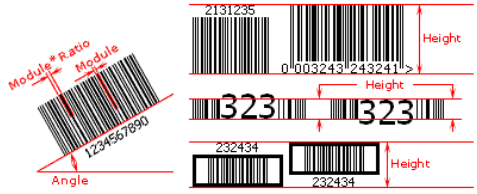
  Note: If the parameter is less than zero, its absolute value specifies the height in pixels in the vertical direction.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to 0 if the Angle is not provided, meaning left to right horizontal direction.

Return:

- If the method succeeds, the return value is zero.
- If the Barcode string length is invalid, the return value is -1.
- If the Ratio parameter is less than or equal to zero, the return value is -2.
- If there is any invalid character in the Barcode string, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character.

See diagram:



Note:

For Delphi 3, the method overload and default value of parameter aren't supported, so the method name is changed to CopyToClipboard2, and the parameters Module, Height, and Angle are required.

B.1.3.3 CopyToClipboard - Syntax 3

Copies a barcode symbol to the system clipboard. The barcode symbol is specified in the parameters of this method.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TDisplayText = (dtNone, dtBarcode, dtFullEncoded);
  { Defined in the pCore1D unit }
  TTextPosition = (tpTopIn, tpTopOut, tpBottomIn, tpBottomOut);
  { Defined in the pCore1D unit }
  TTextAlignment = (taLeft, taCenter, taRight, taJustify, taLeftQuietZone, taCenterQuietZone, taRightQuietZone,
    taJustifyQuietZone, taCustom);
function CopyToClipboard(Data: AnsiString; AutoCheckDigit: Boolean; BarColor, SpaceColor: TColor; BarcodeTextDefine:
  TBarcodeTextDefine; Ratio: Double; Module: Integer = 0; Height: Integer = 0; Angle: Integer = 0): Integer; overload;
  virtual;
```

Description:

Copies a barcode symbol that is specified in the parameters of this method to the system clipboard.

Parameters:

- **Data:** AnsiString; Specifies the barcode text. It is of type AnsiString, so you can specifiy the barcode text in AnsiString format. The method is available only for the Delphi/C++ Builder 2009 or later.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, you can use the method to encode a block of binary (bytes) data under Delphi/C++ Builder 2009 or later. If you want to encode a block of binary (bytes) data into a barcode symbol under Delphi/C++ Builder 2007 or early, please use the CopyToClipboard (Syntax 2) overloading method. Note, the "\" character is used as a escape prefix, so if you want to encode the "\" character, please use the "\\" instead of it.

  For the TBarcode1D_OneCode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

  For the TBarcode1D_FIM component, it is single character that denotes the FIM type, form "A" to "E", the "E" character dnotes an empty barcode symbol.

  For the TBarcode1D_Patch component, it is single character that denotes the Ptach type, and it can be set to "0", "1", "2", "3", "4", "6", and "T", the "0" character dnotes an empty barcode symbol.

  For the TBarcode1D_Code32 component, First "A" character does not need to be entered in the parameter. Also, for the TBarcode1D_PZN component, First "PZN"characters do

not need to be entered in the parameter.

See also the "Data" property.

- **AutoCheckDigit:** Boolean; Specifies whether the check digit should be automatically appended to the barcode symbol.

  See also the "AutoCheckDigit" property.

- **BarColor:** TColor; Specifies the color for all bars in the barcode symbol.

  See also the "BarColor" property.

- **SpaceColor:** TColor; Specifies the color for all spaces in the barcode symbol.

  If the human readable text is represent, the color will be used as its background color (the foreground color is specified using the TextFont field of the BarcodeTextDefine parameter). For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 components, the left spacing and the right spacing will be represented using the color. For TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_UPCA, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones, left quietzone spacing and right quiet zone spacing will be represented using the color. In other words, the parameter specify the background color for entire barcode symbol.

  See also the "SpaceColor" property.

- **BarcodeTextDefine:** TBarcodeTextDefine; Specifies whether to display the human readable text and how todisplay the human readable text. The record is defined in the pBarcode1D unit.

  See also the TBarcodeTextDefine record.

- **Ratio:** Double; Specifies ratio between a wide bar (or space) and a narrow bar (or space) in the barcode symbol. The normal values are from 2.0 to 3.0. If the parameter is less than or equal to zero, the method fails, and the return value is -2.

  See also the "Ratio" property.

- **Module:** Integer; Specifies the module width in logical dots or pixels in the horizontal direction, itis the width of the smallest bar (or space) in the barcode symbol. If the parameter isn't provided or it is set to zero, the value of "Module" property will be used.

  See also the "Module" property.

- **Height:** Integer; Specifies the distance between the top and bottom of a barcode symbol in modules. Ifthe human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included.

  If the parameter isn't provided or its value is set to zero, the value of Height property will be used.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of the top and bottom bearer bars (BearerWidth) are included too.

  If the human readable text is displayed, and it exceeds the barcode symbol in vertical direction, the excess isn't included.

  See also the "Height" property.

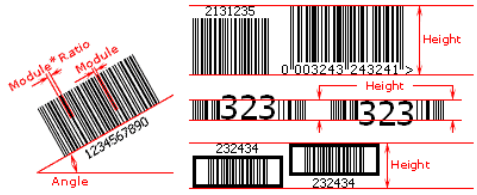
  Note: If the parameter is less than zero, its absolute value specifies the height in pixels in the vertical direction.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to 0 if the Angle is not provided, meaning left to right horizontal direction.

Return:

- If the method succeeds, the return value is zero.
- If the Barcode string length is invalid, the return value is -1.
- If the Ratio parameter is less than or equal to zero, the return value is -2.
- If there is any invalid character in the Barcode string, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character.

See diagram:



Note:

The overloading method is available only for the Delphi/C++ Builder 2009 or later.

B.1.4 Create

Creates and initializes a barcode component.

Syntax:

```
constructor Create(Owner: TComponent); override;
```

Description:

Call Create to instantiate a barcode object at runtime. Barcode components added at design time are created automatically.

Parameters:

- **Owner:** TComponent; It is the component that is responsible for freeing the barcode instance. Typically, this is the form. It becomes the value of the Owner property.

B.1.5 Destroy

Disposes of the instance of the barcode object.

Syntax:

```
destructor Destroy; override;
```

Description:

Destroy is the destructor for a barcode object.

Do not call the destructor directly in an application. Instead, call Free. Free verifies that the barcode object is not nil before it calls Destroy.

B.1.6 Draw

Redraws current barcode symbol in the TImage, TQRImage, or TQRGzImage control that's specified by the Image property.

Syntax:

```
function Draw: Boolean; virtual;
```

Description:

The method redraws the barcode symbol that is specified in the barcode component to the TImage, TQRImage, or TQRGzImage control that's specified by the Image property.

Return:

- If the method succeeds, the return value is true.
- If the Image property is not set, the return value is false.

  If the length of Barcode or Data (only for Delphi/C++ Builder 2009 or later) property value is invalid, the return value is false, corresponding to the OnInvalidLength or OnInvalidDataLength (only for Delphi/C++ Builder 2009 or later) event will occur.

  If there is any invalid character in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property value, the return value is false, corresponding to the OnInvalidChar or OnInvalidDataChar (only for Delphi/C++ Builder 2009 or later) event will occur.

B.1.7 DrawTo

Draws a barcode symbol on the specified canvas. There are several different overloading methods, Syntax 1, Syntax 2, and Syntax 3 (only for Delphi/C++Builder 2009 or later):

- **Syntax 1:** Draws the barcode symbol that is specified in the properties of this barcode component.
- **Syntax 2:** Draws the barcode symbol that is specified in the parameters of this method. The barcode text is specified in the Barcode parameter. It is of type string. For Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiString. For Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, if you use them in the Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiSting in ANSI encoding scheme. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or specify the converted string in the Barcode paramater. Also, you can use the method to encode a block of binary (bytes) data; If you use them in the Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString. By default, the unicode string will be converted to an ANSI encoding string, then be encoded into the barcode symbol. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or use the DrawTo (Syntax 3) overloading method and specify the converted string in its Data paramater. If you want to encode block of binary (bytes) data, please use the DrawTo (Syntax 3) overloading method.
- **Syntax 3:** Draws the barcode symbol that is specified in the parameters of this method. The barcode text is specified in the Data parameter. It is of type AnsiString, so you can specifiy the barcode text in AnsiString format. The method is available only for the Delphi/C++ Builder 2009 or later.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, you can use the method to encode a block of binary (bytes) data under Delphi/C++ Builder 2009 or later. If you want to encode a block of binary (bytes) data into a barcode symbol under Delphi/C++ Builder 2007 or early, please use the DrawTo (Syntax 2) overloading method.

Note:

For Delphi 3, the method overload isn't supported, so the method names of Syntax 1 and Syntax 2 are changed to DrawTo1 and DrawTo2.

B.1.7.1 DrawTo - Syntax 1

Draws a barcode symbol on the specified canvas. The barcode symbol is specified in the properties of this barcode component.

Syntax:

```
function DrawTo(Canvas: TCanvas; Left, Top: Integer; Module: Integer = 0; Height: Integer = 0; Angle: Integer = -1; HDPI:
  Integer = 0; VDPI: Integer = 0): Integer; overload; virtual;
```

Description:

On the specified canvas, draws current barcode symbol that is specified in the properties of this barcode component.

Parameters:

- **Canvas:** TCanvas; Specifies target canvas to represent the barcode symbol in it.
- **Left:** Integer; Specifies the margin between the barcode symbol and the left side of the canvas in logical dots or pixels in the horizontal direction. If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too.

  See also the "LeftMargin" property.
- **Top:** Integer; Specifies the margin between the barcode symbol and the top side of the canvas in logical dots or pixels in the vertical direction. If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too.

  See also the "TopMargin" property.
- **Module:** Integer; Specifies the module width in logical dots or pixels in the horizontal direction, it is the width of the smallest bar (or space) in the barcode symbol. If the parameter isn't provided or it is set to zero, the value of "Module" property will be used.

  See also the "Module" property.
- **Height:** Integer; Specifies the distance between the top and bottom of a barcode symbol in modules. If the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included.

  If the parameter isn't provided or its value is set to zero, the value of Height property will be used.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of the top and bottom bearer bars (BearerWidth) are included too.

  If the human readable text is displayed, and it exceeds the barcode symbol in vertical direction, the excess isn't included.

  See also the "Height" property.

  Note: If the parameter is less than zero, its absolute value specifies the height in logical dots or pixels in the vertical direction.
- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to -1 if theAngle is not provided, and the barcode symbol will be rotated base on the value of the Orientation property:
  - boLeftRight: 0 degrees
  - boRightLeft: 180 degrees
  - boTopBottom: 270 degrees
  - boBottomTop: 90 degrees

  If you want to use the -1 degrees, the 359 degrees can be used instead.
- **HDPI:** Integer, Specifies the horizontal resolution of canvas in logical DPI. It's the number of logical dots or pixels per inch.

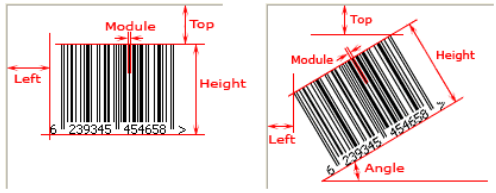
  It defaults to 0 if the HDPI is not provided. If it is set to less than or equal to zero, the physical horizontal resolution obtained from the Canvas parameter will be used. So if you use the MM_TEXT map mode, you can specify it to 0. If you use the MM_ISOTROPIC or MM_ANISOTROPIC map mode, and the horizontal units or scaling is changed, the parameter required in order to represent the correct barcode symbol when the symbol is rotated.
- **VDPI:** Integer, Specifies the vertical resolution of canvas in logical DPI. It's the number of logical dots or pixels per inch.

  It defaults to 0 if the VDPI is not provided. If it is set to less than or equal to zero, the physical vertical resolution obtained from the Canvas parameter will be used. So if you use the MM_TEXT map mode, you can specify it to 0. If you use the MM_ISOTROPIC or MM_ANISOTROPIC map mode, and the vertical units or scaling is changed, the parameter required in order to represent the correct barcode symbol when the symbol is rotated.

Return:

- If the method succeeds, the return value is zero.
- If the length of the barcode text that is specified by Barcode or Data (only for Delphi/C++ Builder 2009 or later)property is invalid, the return value is -1. Corresponding to the OnInvalidLength or OnInvalidDataLength (only for Delphi/C++ Builder 2009 or later) event will occur.
- If there is any invalid character in the the barcode text that is specified by Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character. Corresponding to the OnInvalidChar or OnInvalidDataChar (only for Delphi/C++ Builder 2009 or later) event will occur. For the TBarcode1D_OneCode component, if the invalid character in the Tracking property, the index is from 1 to 20 including 1 and 20; if it is in the Routing property, the value starts with 21 (First character of the Routing property).

See diagram:



Note:

For Delphi 3, the method overload and default value of parameter aren't supported, so the method name is changed to DrawTo1, and the parameters Module, Height, Angle, HDPI, and VDPI are required.

B.1.7.2 DrawTo - Syntax 2

Draws a barcode symbol on the specified canvas. The barcode symbol is specified in the parameters of this method.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TDisplayText = (dtNone, dtBarcode, dtFullEncoded);
  { Defined in the pCore1D unit }
  TTextPosition = (tpTopIn, tpTopOut, tpBottomIn, tpBottomOut);
  { Defined in the pCore1D unit }
  TTextAlignment = (taLeft, taCenter, taRight, taJustify, taLeftQuietZone, taCenterQuietZone, taRightQuietZone,
    taJustifyQuietZone, taCustom);
function DrawTo(Canvas: TCanvas; Left, Top: Integer; Barcode: string; AutoCheckDigit: Boolean; BarColor, SpaceColor: TColor;
  BarcodeTextDefine: TBarcodeTextDefine; Ratio: Double; Module: Integer = 0; Height: Integer = 0; Angle: Integer = 0; HDPI:
  Integer = 0; VDPI: Integer = 0): Integer; overload; virtual;
```

Description:

On the specified canvas, draws a barcode symbol that is specified in the parameters of this method.

Parameters:

- **Canvas:** TCanvas; Specifies target canvas to represent the barcode symbol in it.
- **Left:** Integer; Specifies the margin between the barcode symbol and the left side of the canvas in logical dots or pixels in the horizontal direction. If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too.

  See also the "LeftMargin" property.
- **Top:** Integer; Specifies the margin between the barcode symbol and the top side of the canvas in logical dots or pixels in the vertical direction. If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too.

  See also the "TopMargin" property.
- **Barcode:** String; Specifies the barcode text. It is of type string. For Delphi/C++ Builder 2007 or early, theBarcode parameter is in fact an AnsiString. For Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString.

For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, if you use them in the Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiSting in ANSI encoding scheme. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or specify the converted string in the Barcode paramater. Also, you can use the method to encode a block of binary (bytes) data; If you use them in the Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString. By default, the unicode string will be converted to an ANSI encoding string, then be encoded into the barcode symbol. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or use the DrawTo (Syntax 3) overloading method and specify the converted string in its Data paramater. If you want to encode block of binary (bytes) data, please use the DrawTo (Syntax 3) overloading method. Note, the "\" character is used as a escape prefix, so if you want to encode the "\" character, please use the "\\" instead of it.

For the TBarcode1D_OneCode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

For the TBarcode1D_FIM component, it is single character that denotes the FIM type, form "A" to "E", the "E" character dnotes an empty barcode symbol.

For the TBarcode1D_Patch component, it is single character that denotes the Ptach type, and it can be set to "0", "1", "2", "3", "4", "6", and "T", the "0" character dnotes an empty barcode symbol.

For the TBarcode1D_Code32 component, First "A" character does not need to be entered in the parameter. Also, for the TBarcode1D_PZN component, First "PZN" characters do not need to be entered in the parameter.

See also the "Barcode" property.

- **AutoCheckDigit:** Boolean; Specifies whether the check digit should be automatically appended to the barcode symbol.

  See also the "AutoCheckDigit" property.

- **BarColor:** TColor; Specifies the color for all bars in the barcode symbol.

  See also the "BarColor" property.

- **SpaceColor:** TColor; Specifies the color for all spaces in the barcode symbol.

  If the human readable text is represent, the color will be used as its background color (the foreground color is specified using the TextFont field of the BarcodeTextDefine parameter). For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 components, the left spacing and the right spacing will be represented using the color. For TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_UPCA, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones, left quietzone spacing and right quiet zone spacing will be represented using the color. In other words, the parameter specify the background color for entire barcode symbol.

  See also the "SpaceColor" property.

- **BarcodeTextDefine:** TBarcodeTextDefine; Specifies whether to display the human readable text and how to display the human readable text. The record is defined in the pBarcode1D unit.

  See also the TBarcodeTextDefine record.

- **Ratio:** Double; Specifies ratio between a wide bar (or space) and a narrow bar (or space) in the barcode symbol. The normal values are from 2.0 to 3.0. If the parameter is less than or equal to zero, the method fails, and the return value is -2.

  See also the "Ratio" property.

- **Module:** Integer; Specifies the module width in logical dots or pixels in the horizontal direction, it is the width of the smallest bar (or space) in the barcode symbol. If the parameter isn't provided or it is set to zero, the value of "Module" property will be used.

  See also the "Module" property.

- **Height:** Integer; Specifies the distance between the top and bottom of a barcode symbol in modules. If the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included.

  If the parameter isn't provided or its value is set to zero, the value of Height property will be used.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of the top and bottom bearer bars (BearerWidth) are included too.

  If the human readable text is displayed, and it exceeds the barcode symbol in vertical direction, the excess isn't included.

  See also the "Height" property.

  Note: If the parameter is less than zero, its absolute value specifies the height in logical dots or pixels in the vertical direction.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to 0 if the Angle is not provided, meaning left to right horizontal direction.

- **HDPI:** Integer, Specifies the horizontal resolution of canvas in logical DPI. It's the number of logical dots or pixels per inch.

  It defaults to 0 if the HDPI is not provided. If it is set to less than or equal to zero, the physical horizontal resolution obtained from the Canvas parameter will be used. So if you use the MM_TEXT map mode, you can specify it to 0. If you use the MM_ISOTROPIC or MM_ANISOTROPIC map mode, and the horizontal units or scaling is changed, the parameter required in order to represent the correct barcode symbol when the symbol is rotated.

- **VDPI:** Integer, Specifies the vertical resolution of canvas in logical DPI. It's the number of logical dots or pixels per inch.

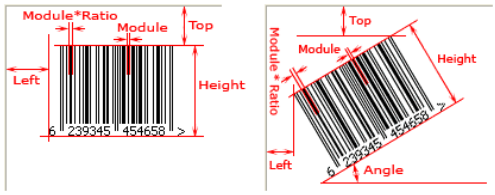
  It defaults to 0 if the VDPI is not provided. If it is set to less than or equal to zero, the physical vertical resolution obtained from the Canvas parameter will be used. So if you use the MM_TEXT map mode, you can specify it to 0. If you use the MM_ISOTROPIC or MM_ANISOTROPIC map mode, and the vertical units or scaling is changed, the parameter required in order to represent the correct barcode symbol when the symbol is rotated.

Return:

- If the method succeeds, the return value is zero.
- If the Barcode string length is invalid, the return value is -1.
- If the Ratio parameter is less than or equal to zero, the return value is -2.

- If there is any invalid character in the Barcode string, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character.

See diagram:



Note:

For Delphi 3, the method overload and default value of parameter aren't supported, so the method name is changed to DrawTo2, and the parameters Module, Height, Angle, HDPI, and VDPI are required.

B.1.7.3 DrawTo - Syntax 3

Draws a barcode symbol on the specified canvas. The barcode symbol is specified in the parameters of this method.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TDisplayText = (dtNone, dtBarcode, dtFullEncoded);
  { Defined in the pCore1D unit }
  TTextPosition = (tpTopIn, tpTopOut, tpBottomIn, tpBottomOut);
  { Defined in the pCore1D unit }
  TTextAlignment = (taLeft, taCenter, taRight, taJustify, taLeftQuietZone, taCenterQuietZone, taRightQuietZone,
    taJustifyQuietZone, taCustom);
function DrawTo(Canvas: TCanvas; Left, Top: Integer; Data: AnsiString; AutoCheckDigit: Boolean; BarColor, SpaceColor: TColor;
  BarcodeTextDefine: TBarcodeTextDefine; Ratio: Double; Module: Integer = 0; Height: Integer = 0; Angle: Integer = 0; HDPI:
  Integer = 0; VDPI: Integer = 0): Integer; overload; virtual;
```

Description:

On the specified canvas, draws a barcode symbol that is specified in the parameters of this method.

Parameters:

- **Canvas:** TCanvas; Specifies target canvas to represent the barcode symbol in it.
- **Left:** Integer; Specifies the margin between the barcode symbol and the left side of the canvas in logical dots or pixels in the horizontal direction. If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too.

  See also the "LeftMargin" property.

- **Top:** Integer; Specifies the margin between the barcode symbol and the top side of the canvas in logical dots or pixels in the vertical direction. If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too.

  See also the "TopMargin" property.

- **Data:** AnsiString; Specifies the barcode text. It is of type AnsiString, so you can specifiy the barcode text in AnsiString format. The method is available only for the Delphi/C++ Builder 2009 or later.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, you can use the method to encode a block of binary (bytes) data under Delphi/C++ Builder 2009 or later. If you want to encode a block of binary (bytes) data into a barcode symbol under Delphi/C++ Builder 2007 or early, please use the DrawTo (Syntax 2) overloading method. Note, the "\" character is used as a escape prefix, so if you want to encode the "\" character, please use the "\\" instead of it.

  For the TBarcode1D_OneCode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

  For the TBarcode1D_FIM component, it is single character that denotes the FIM type, form "A" to "E", the "E" character dnotes an empty barcode symbol.

For the TBarcode1D_Patch component, it is single character that denotes the Ptach type, and it can be set to "0", "1", "2", "3", "4", "6", and "T", the "0" character dnotes an empty barcode symbol.

For the TBarcode1D_Code32 component, First "A" character does not need to be entered in the parameter. Also, for the TBarcode1D_PZN component, First "PZN" characters do not need to be entered in the parameter.

See also the "Data" property.

- **AutoCheckDigit:** Boolean; Specifies whether the check digit should be automatically appended to the barcode symbol.

  See also the "AutoCheckDigit" property.

- **BarColor:** TColor; Specifies the color for all bars in the barcode symbol.

  See also the "BarColor" property.

- **SpaceColor:** TColor; Specifies the color for all spaces in the barcode symbol.

  If the human readable text is represent, the color will be used as its background color (the foreground color is specified using the TextFont field of the BarcodeTextDefine parameter). For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 components, the left spacing and the right spacing will be represented using the color. For TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_UPCA, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones, left quietzone spacing and right quiet zone spacing will be represented using the color. In other words, the parameter specify the background color for entire barcode symbol.

  See also the "SpaceColor" property.

- **BarcodeTextDefine:** TBarcodeTextDefine; Specifies whether to display the human readable text and how to display the human readable text. The record is defined in the pBarcode1D unit.

  See also the TBarcodeTextDefine record.

- **Ratio:** Double; Specifies ratio between a wide bar (or space) and a narrow bar (or space) in the barcode symbol. The normal values are from 2.0 to 3.0. If the parameter is less than or equal to zero, the method fails, and the return value is -2.

  See also the "Ratio" property.

- **Module:** Integer; Specifies the module width in logical dots or pixels in the horizontal direction, it is the width of the smallest bar (or space) in the barcode symbol. If the parameter isn't provided or it is set to zero, the value of "Module" property will be used.

  See also the "Module" property.

- **Height:** Integer; Specifies the distance between the top and bottom of a barcode symbol in modules. If the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included.

  If the parameter isn't provided or its value is set to zero, the value of Height property will be used.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of the top and bottom bearer bars (BearerWidth) are included too.

  If the human readable text is displayed, and it exceeds the barcode symbol in vertical direction, the excess isn't included.

  See also the "Height" property.

  Note: If the parameter is less than zero, its absolute value specifies the height in logical dots or pixels in the vertical direction.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to 0 if the Angle is not provided, meaning left to right horizontal direction.

- **HDPI:** Integer, Specifies the horizontal resolution of canvas in logical DPI. It's the number of logical dots or pixels per inch.

  It defaults to 0 if the HDPI is not provided. If it is set to less than or equal to zero, the physical horizontal resolution obtained from the Canvas parameter will be used. So if you use the MM_TEXT map mode, you can specify it to 0. If you use the MM_ISOTROPIC or MM_ANISOTROPIC map mode, and the horizontal units or scaling is changed, the parameter required in order to represent the correct barcode symbol when the symbol is rotated.

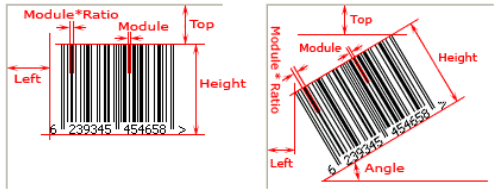
- **VDPI:** Integer, Specifies the vertical resolution of canvas in logical DPI. It's the number of logical dots or pixels per inch.

  It defaults to 0 if the VDPI is not provided. If it is set to less than or equal to zero, the physical vertical resolution obtained from the Canvas parameter will be used. So if you use the MM_TEXT map mode, you can specify it to 0. If you use the MM_ISOTROPIC or MM_ANISOTROPIC map mode, and the vertical units or scaling is changed, the parameter required in order to represent the correct barcode symbol when the symbol is rotated.

Return:

- If the method succeeds, the return value is zero.
- If the Barcode string length is invalid, the return value is -1.
- If the Ratio parameter is less than or equal to zero, the return value is -2.
- If there is any invalid character in the Barcode string, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character.

See diagram:



Note:

The overloading method is available only for the Delphi/C++ Builder 2009 or later.

B.1.8 DrawToSize

Returns the horizontal width and vertical height of a rotated barcode symbol in pixels. There are several different overloading methods, Syntax 1, Syntax 2, and Syntax 3 (only for Delphi/C++Builder 2009 or later):

- **Syntax 1:** Returns the horizontal width and vertical height of the rotated barcode symbol that is specified by the properties of this barcode component.
- **Syntax 2:** Returns the horizontal width and vertical height of the rotated barcode symbol that is specified by the parameters of this method. The barcode text is specified in the Barcode parameter. It is of type string. For Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact a AnsiString. For Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, if you use them in the Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiSting in ANSI encoding scheme. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or specify the converted string in the Barcode paramater. Also, you can use the method if you encode a block of binary (bytes) data; If you use them in the Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString. By default, the unicode string will be converted to an ANSI encoding string, then be encoded into the barcode symbol. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or use the DrawToSize (Syntax 3) overloading method and specify the converted string in its Data paramater. If you encode a block of binary (bytes) data, please use the DrawToSize (Syntax 3) overloading method.
- **Syntax 3:** Returns the horizontal width and vertical height of the rotated barcode symbol that is specified by the parameters of this method. The barcode text is specified in the Data parameter. It is of type AnsiString, so you can specifiy the barcode text in AnsiString format. The method is available only for the Delphi/C++ Builder 2009 or later.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, you can use the method if you encode a block of binary (bytes) data under Delphi/C++ Builder 2009 or later. If you encode a block of binary (bytes) data into a barcode symbol under Delphi/C++ Builder 2007 or early, please use the DrawToSize (Syntax 2) overloading method.

Note:

For Delphi 3, the method overload isn't supported, so the method names of Syntax 1 and Syntax 2 are changed to DrawToSize1 and DrawToSize2.

B.1.8.1 DrawToSize - Syntax 1

Returns the horizontal width and vertical height of a rotated barcode symbol in logical dots or pixels. The barcode symbol is specified in the properties of this barcode component.

Syntax:

```
function DrawToSize(var TotalWidth, TotalHeight, SymbolWidth, SymbolHeight: Integer; Canvas: TCanvas; Module: Integer = 0;
  Height: Integer = 0; Angle: Integer = -1; HDPI: Integer = 0; VDPI: Integer = 0): Integer; overload; virtual;
```

Description:

The method returns the horizontal width and vertical height of the rotated barcode symbol that is specified by properties of this barcode component, in logical dots or pixels.

Parameters:

- **TotalWidth:** Integer; Returns the horizontal width of the rotated barcode symbol in logical dots or pixels in the horizontal direction.

  Before rotation, if the human readable text is represented, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.
- **TotalHeight:** Integer; Returns the vertical height of the rotated barcode symbol in logical dots or pixels in the vertical direction.

  Before rotation, if the human readable text is represented, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and

TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolWidth:** Integer; Returns the distance between the leading and trailing of the rotated barcode symbol in logical dots or pixels in the horizontal direction.

  Before rotation, if the human readable text is represented, the width of human readable text will be consulted. Otherwise, it will not be consulted. Note, if the human readable text is represented, and it exceeds the barcode symbol in horizontal direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolHeight:** Integer; Returns the distance between the top and bottom of the rotated barcode symbol in logical dots or pixels in the vertical direction.

  Before rotation, if the human readable text is represented, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess is included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars (BearerWidth) are included too.

  See also the "Height" property.

- **Canvas:** TCanvas; Specifies target canvas that the barcode symbol will be represented in it. If the DisplayText property isn't set to dtNone, the parameter is required in order to calculate the width of human readable text. Otherwise, it will be ignored, so you can set it to nil.

- **Module:** Integer; Specifies the module width in logical dots or pixels in the horizontal direction, it is the width of the smallest bar (or space) in the barcode symbol. If the parameter isn't provided or it is set to zero, the value of "Module" property will be used.

  See also the "Module" property.

- **Height:** Integer; Specifies the distance between the top and bottom of a barcode symbol in modules. If the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included.

  If the parameter isn't provided or its value is set to zero, the value of Height property will be used.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of the top and bottom bearer bars (BearerWidth) are included too.

  If the human readable text is displayed, and it exceeds the barcode symbol in vertical direction, the excess isn't included.

  See also the "Height" property.

  Note: If the parameter is less than zero, its absolute value specifies the height in logical dots or pixels in the vertical direction.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to -1 if the Angle is not provided, and the barcode symbol will be rotated base on the value of the Orientation property:

  - boLeftRight: 0 degrees
  - boRightLeft: 180 degrees
  - boTopBottom: 270 degrees
  - boBottomTop: 90 degrees

  If you want to use the -1 degrees, the 359 degrees can be used instead.

- **HDPI:** Integer, Specifies the horizontal resolution of canvas in logical DPI. It's the number of logical dots or pixels per inch.

  It defaults to 0 if the HDPI is not provided. If it is set to less than or equal to zero, the physical horizontal resolution obtained from the Canvas parameter will be used. So if you use the MM_TEXT map mode, you can specify it to 0. If you use the MM_ISOTROPIC or MM_ANISOTROPIC map mode, and the horizontal units or scaling is changed, the parameter required in order to obtain the correct sizes when the symbol is rotated.

  When the Canvas parameter is set to nil, if both HDPI and VDPI are set to 0, it indicates the horizontal resolution is equal to the vertical resolution.

- **VDPI:** Integer, Specifies the vertical resolution of canvas in logical DPI. It's the number of logical dots or pixels per inch.

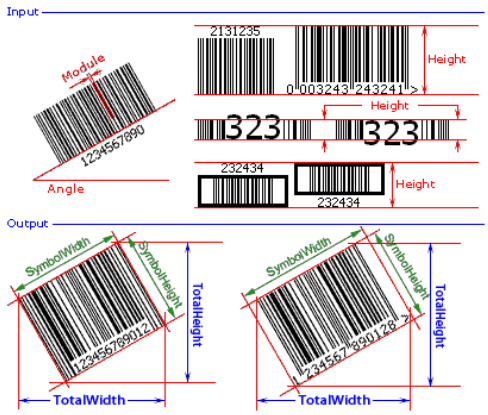
  It defaults to 0 if the VDPI is not provided. If it is set to less than or equal to zero, the physical vertical resolution obtained from the Canvas parameter will be used. So if you use the MM_TEXT map mode, you can specify it to 0. If you use the MM_ISOTROPIC or MM_ANISOTROPIC map mode, and the vertical units or scaling is changed, the parameter required in order to obtain the correct sizes when the symbol is rotated.

  When the Canvas parameter is set to nil, if both HDPI and VDPI are set to 0, it indicates the horizontal resolution is equal to the vertical resolution.

Return:

- If the method succeeds, the return value is zero.
- If the length of the barcode text that is specified by Barcode or Data (only for Delphi/C++ Builder 2009 or later)property is invalid, the return value is -1. Corresponding to the OnInvalidLength or OnInvalidDataLength (only for Delphi/C++ Builder 2009 or later) event will occur.
- If there is any invalid character in the the barcode text that is specified by Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character. Corresponding to the OnInvalidChar or OnInvalidDataChar (only for Delphi/C++ Builder 2009 or later) event will occur. For the TBarcode1D_OneCode component, if the invalid character in the Tracking property, the index is from 1 to 20 including 1 and 20; if it is in the Routing property, the value starts with 21 (First character of the Routing property).

See diagram:



Note:

For Delphi 3, the method overload and default value of parameter aren't supported, so the method name is changed to DrawToSize1, and the parameters Module, Height, Angle, HDPI, and VDPI are required.

B.1.8.2 DrawToSize - Syntax 2

Returns the horizontal width and vertical height of a rotated barcode symbol in logical dots or pixels. The barcode symbol is specified in the parameters of this method.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TDisplayText = (dtNone, dtBarcode, dtFullEncoded);
  { Defined in the pCore1D unit }
  TTextAlignment = (taLeft, taCenter, taRight, taJustify, taLeftQuietZone, taCenterQuietZone, taRightQuietZone,
    taJustifyQuietZone, taCustom);
function DrawToSize(var TotalWidth, TotalHeight, SymbolWidth, SymbolHeight: Integer; Canvas: TCanvas; Barcode: string;
  AutoCheckDigit: Boolean; BarcodeTextDefine: TBarcodeTextDefine; Ratio: Double; Module: Integer = 0; Height: Integer = 0;
  Angle: Integer = 0; HDPI: Integer = 0; VDPI: Integer = 0): Integer; overload; virtual;
```

Description:

The method returns the horizontal width and vertical height of the rotated barcode symbol that is specified by parameters of this method, in logical dots or pixels.

Parameters:

- **TotalWidth:** Integer; Returns the horizontal width of the rotated barcode symbol in logical dots or pixels in the horizontal direction.

  Before rotation, if the human readable text is represented, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **TotalHeight:** Integer; Returns the vertical height of the rotated barcode symbol in logical dots or pixels in the vertical direction.

  Before rotation, if the human readable text is represented, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolWidth:** Integer; Returns the distance between the leading and trailing of the rotated barcode symbol in logical dots or pixels in the horizontal direction.

  Before rotation, if the human readable text is represented, the width of human readable text will be consulted. Otherwise, it will not be consulted. Note, if the human readable text is represented, and it exceeds the barcode symbol in horizontal direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and

TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolHeight:** Integer; Returns the distance between the top and bottom of the rotated barcode symbol in logical dots or pixels in the vertical direction.

  Before rotation, if the human readable text is represented, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess is included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars (BearerWidth) are included too.

  See also the "Height" property.

- **Canvas:** TCanvas; Specifies target canvas that the barcode symbol will be represented in it. If theDisplayText property isn't set to dtNone, the parameter is required in order to calculate the width of human readable text. Otherwise, it will be ignored, so you can set it to nil.

- **Barcode:** String; Specifies the barcode text. It is of type string. For Delphi/C++ Builder 2007 or early, theBarcode parameter is in fact an AnsiString. For Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, if you use them in the Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiSting in ANSI encoding scheme. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in theOnEncode event, or specify the converted string in the Barcode paramater. Also, you can use the method if you encode a block of binary (bytes) data; If you use them in the Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString. By default, the unicode string will be converted to an ANSI encoding string, then be encoded into the barcode symbol. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in theOnEncode event, or use the DrawToSize (Syntax 3) overloading method and specify the converted string in its Data paramater. If you encode a block of binary (bytes) data, please use the DrawToSize (Syntax 3) overloading method. Note, the "\" character is used as a escape prefix, so if you want to encode the "\" character, please use the "\\" instead of it.

  For the TBarcode1D_OneCode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

  For the TBarcode1D_FIM component, it is single character that denotes the FIM type, form "A" to "E", the "E" character dnotes an empty barcode symbol.

  For the TBarcode1D_Patch component, it is single character that denotes the Ptach type, and it can be set to "0", "1", "2", "3", "4", "6", and "T", the "0" character dnotes an empty barcode symbol.

  For the TBarcode1D_Code32 component, First "A" character does not need to be entered in the parameter. Also, for the TBarcode1D_PZN component, First "PZN" characters do not need to be entered in the parameter.

  See also the "Barcode" property.

- **AutoCheckDigit:** Boolean; Specifies whether the check digit should be automatically appended to the barcode symbol.

  See also the "AutoCheckDigit" property.

- **BarcodeTextDefine:** TBarcodeTextDefine; Specifies whether to display the human readable text and how to display the human readable text. The record is defined in the pBarcode1D unit.

  See also the TBarcodeTextDefine record.

- **Ratio:** Double; Specifies ratio between a wide bar (or space) and a narrow bar (or space) in the barcode symbol. The normal values are from 2.0 to 3.0. If the parameter is less than or equal to zero, the method fails, and the return value is -2.

  See also the "Ratio" property.

- **Module:** Integer; Specifies the module width in logical dots or pixels in the horizontal direction, it is the width of the smallest bar (or space) in the barcode symbol. If the parameter isn't provided or it is set to zero, the value of "Module" property will be used.

  See also the "Module" property.

- **Height:** Integer; Specifies the distance between the top and bottom of a barcode symbol in modules. If the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included.

  If the parameter isn't provided or its value is set to zero, the value of Height property will be used.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of the top and bottom bearer bars (BearerWidth) are included too.

  If the human readable text is displayed, and it exceeds the barcode symbol in vertical direction, the excess isn't included.

  See also the "Height" property.

  Note: If the parameter is less than zero, its absolute value specifies the height in logical dots or pixels in the vertical direction.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to 0 if the Angle is not provided, meaning left to right horizontal direction.

- **HDPI:** Integer, Specifies the horizontal resolution of canvas in logical DPI. It's the number of logical dots or pixels per inch.

  It defaults to 0 if the HDPI is not provided. If it is set to less than or equal to zero, the physical horizontal resolution obtained from the Canvas parameter will be used. So if you use the MM_TEXT map mode, you can specify it to 0. If you use the MM_ISOTROPIC or MM_ANISOTROPIC map mode, and the horizontal units or scaling is changed, the parameter required in order to obtain the correct sizes when the symbol is rotated.

  When the Canvas parameter is set to nil, if both HDPI and VDPI are set to 0, it indicates the horizontal resolution is equal to the vertical resolution.

- **VDPI:** Integer, Specifies the vertical resolution of canvas in logical DPI. It's the number of logical dots or pixels per inch.

  It defaults to 0 if the VDPI is not provided. If it is set to less than or equal to zero, the physical vertical resolution obtained from the Canvas parameter will be used. So if you use the MM_TEXT map mode, you can specify it to 0. If you use the MM_ISOTROPIC or MM_ANISOTROPIC map mode, and the vertical units or scaling is changed, the parameter

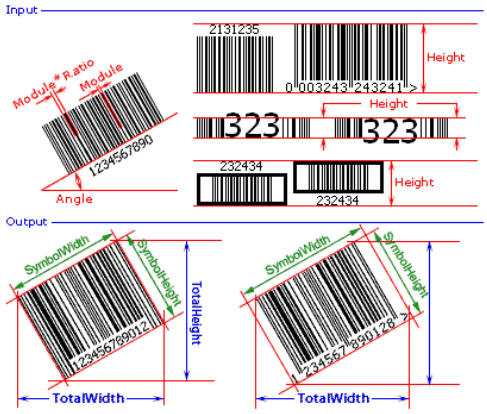
required in order to obtain the correct sizes when the symbol is rotated.

When the Canvas parameter is set to nil, if both HDPI and VDPI are set to 0, it indicates the horizontal resolution is equal to the vertical resolution.

Return:

- If the method succeeds, the return value is zero.
- If the Barcode string length is invalid, the return value is -1.
- If the Ratio parameter is less than or equal to zero, the return value is -2.
- If there is any invalid character in the Barcode string, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character.

See diagram:



Note:

For Delphi 3, the method overload and default value of parameter aren't supported, so the method name is changed to DrawToSize2, and the parameters Module, Height, Angle, HDPI, and VDPI are required.

B.1.8.3 DrawToSize - Syntax 3

Returns the horizontal width and vertical height of a rotated barcode symbol in logical dots or pixels. The barcode symbol is specified in the parameters of this method.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TDisplayText = (dtNone, dtBarcode, dtFullEncoded);
  { Defined in the pCore1D unit }
  TTextAlignment = (taLeft, taCenter, taRight, taJustify, taLeftQuietZone, taCenterQuietZone, taRightQuietZone,
    taJustifyQuietZone, taCustom);
function DrawToSize(var TotalWidth, TotalHeight, SymbolWidth, SymbolHeight: Integer; Canvas: TCanvas; Data: AnsiString;
  AutoCheckDigit: Boolean; BarcodeTextDefine: TBarcodeTextDefine; Ratio: Double; Module: Integer = 0; Height: Integer = 0;
  Angle: Integer = 0; HDPI: Integer = 0; VDPI: Integer = 0): Integer; overload; virtual;
```

Description:

The method returns the horizontal width and vertical height of the rotated barcode symbol that is specified by parameters of this method, in logical dots or pixels.

Parameters:

- **TotalWidth:** Integer; Returns the horizontal width of the rotated barcode symbol in logical dots or pixels in the horizontal direction.

  Before rotation, if the human readable text is represented, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **TotalHeight:** Integer; Returns the vertical height of the rotated barcode symbol in logical dots or pixels in the vertical direction.

  Before rotation, if the human readable text is represented, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolWidth:** Integer; Returns the distance between the leading and trailing of the rotated barcode symbol in logical dots or pixels in the horizontal direction.

  Before rotation, if the human readable text is represented, the width of human readable text will be consulted. Otherwise, it will not be consulted. Note, if the human readable text is represented, and it exceeds the barcode symbol in horizontal direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolHeight:** Integer; Returns the distance between the top and bottom of the rotated barcode symbol in logical dots or pixels in the vertical direction.

  Before rotation, if the human readable text is represented, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess is included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars (BearerWidth) are included too.

  See also the "Height" property.

- **Canvas:** TCanvas; Specifies target canvas that the barcode symbol will be represented in it. If theDisplayText property isn't set to dtNone, the parameter is required in order to calculate the width of human readable text. Otherwise, it will be ignored, so you can set it to nil.

- **Data:** AnsiString; Specifies the barcode text. It is of typeAnsiString, so you can specifiy the barcode text in AnsiString format. The method is available only for the Delphi/C++ Builder 2009 or later.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, you can use the method if you encode a block of binary (bytes) data under Delphi/C++ Builder 2009 or later. If you encode a block of binary (bytes) data into a barcode symbol under Delphi/C++ Builder 2007 or early, please use the DrawToSize (Syntax 2) overloading method. Note, the "\" character is used as a escape prefix, so if you want to encode the "\" character, please use the "\\" instead of it.

  For the TBarcode1D_OneCode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

  For the TBarcode1D_FIM component, it is single character that denotes the FIM type, form "A" to "E", the "E" character dnotes an empty barcode symbol.

  For the TBarcode1D_Patch component, it is single character that denotes the Ptach type, and it can be set to "0", "1", "2", "3", "4", "6", and "T", the "0" character dnotes an empty barcode symbol.

  For the TBarcode1D_Code32 component, First "A" character does not need to be entered in the parameter. Also, for the TBarcode1D_PZN component, First "PZN" characters do not need to be entered in the parameter.

  See also the "Data" property.

- **AutoCheckDigit:** Boolean; Specifies whether the check digit should be automatically appended to the barcode symbol.

  See also the "AutoCheckDigit" property.

- **BarcodeTextDefine:** TBarcodeTextDefine; Specifies whether to display the human readable text and how to display the human readable text. The record is defined in the pBarcode1D unit.

  See also the TBarcodeTextDefine record.

- **Ratio:** Double; Specifies ratio between a wide bar (or space) and a narrow bar (or space) in the barcode symbol. The normal values are from 2.0 to 3.0. If the parameter is less than or equal to zero, the method fails, and the return value is -2.

  See also the "Ratio" property.

- **Module:** Integer; Specifies the module width in logical dots or pixels in the horizontal direction, it is the width of the smallest bar (or space) in the barcode symbol. If the parameter isn't provided or it is set to zero, the value of "Module" property will be used.

  See also the "Module" property.

- **Height:** Integer; Specifies the distance between the top and bottom of a barcode symbol in modules. If the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included.

  If the parameter isn't provided or its value is set to zero, the value of Height property will be used.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of the top and bottom bearer bars (BearerWidth) are included too.

  If the human readable text is displayed, and it exceeds the barcode symbol in vertical direction, the excess isn't included.

  See also the "Height" property.

  Note: If the parameter is less than zero, its absolute value specifies the height in logical dots or pixels in the vertical direction.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to 0 if the Angle is not provided, meaning left to right horizontal direction.

- **HDPI:** Integer, Specifies the horizontal resolution of canvas in logical DPI. It's the number of logical dots or pixels per inch.

  It defaults to 0 if the HDPI is not provided. If it is set to less than or equal to zero, the physical horizontal resolution obtained from the Canvas parameter will be used. So if you use the MM_TEXT map mode, you can specify it to 0. If you use the MM_ISOTROPIC or MM_ANISOTROPIC map mode, and the horizontal units or scaling is changed, the parameter

required in order to obtain the correct sizes when the symbol is rotated.

When the Canvas parameter is set to nil, if both HDPI and VDPI are set to 0, it indicates the horizontal resolution is equal to the vertical resolution.

- **VDPI:** Integer, Specifies the vertical resolution of canvas in logical DPI. It's the number of logical dots or pixels per inch.

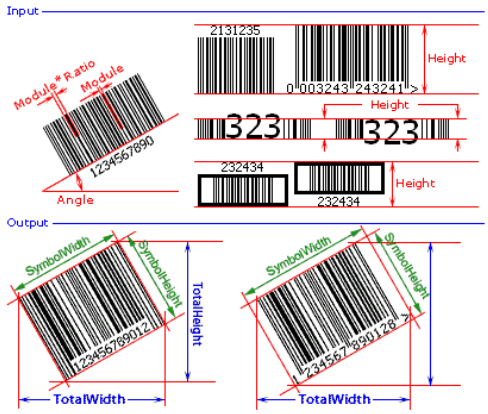
  It defaults to 0 if the VDPI is not provided. If it is set to less than or equal to zero, the physical vertical resolution obtained from the Canvas parameter will be used. So if you use the MM_TEXT map mode, you can specify it to 0. If you use the MM_ISOTROPIC or MM_ANISOTROPIC map mode, and the vertical units or scaling is changed, the parameter required in order to obtain the correct sizes when the symbol is rotated.

  When the Canvas parameter is set to nil, if both HDPI and VDPI are set to 0, it indicates the horizontal resolution is equal to the vertical resolution.

Return:

- If the method succeeds, the return value is zero.
- If the Barcode string length is invalid, the return value is -1.
- If the Ratio parameter is less than or equal to zero, the return value is -2.
- If there is any invalid character in the Barcode string, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character.

See diagram:



Note:

The overloading method is available only for the Delphi/C++ Builder 2009 or later.

B.1.9 Print

Prints specified barcode symbol to printer. There are several different overloading methods, Syntax 1, Syntax 2, and Syntax 3 (only for Delphi/C++Builder 2009 or later):

- **Syntax 1:** Prints the barcode symbol that is specified in the properties of this barcode component.
- **Syntax 2:** Prints the barcode symbol that is specified in the parameters of this method. The barcode text is specified in the Barcode parameter. It is of type string. For Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiString. For Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, if you use them in the Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiSting in ANSI encoding scheme. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or specify the converted string in the Barcode paramater. Also, you can use the method to encode a block of binary (bytes) data; If you use them in the Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString. By default, the unicode string will be converted to an ANSI encoding string, then be encoded into the barcode symbol. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or use the Print (Syntax 3) overloading method and specify the converted string in its Data paramater. If you want to encode block of binary (bytes) data, please use the Print (Syntax 3) overloading method.
- **Syntax 3:** Prints the barcode symbol that is specified in the parameters of this method. The barcode text is specified in the Data parameter. It is of type AnsiString, so you can specifiy the barcode text in AnsiString format. The method is available only for the Delphi/C++ Builder 2009 or later.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, you can use the method to encode a block of binary (bytes) data under Delphi/C++ Builder 2009 or later. If you want to encode a block of binary (bytes) data into a barcode symbol under Delphi/C++ Builder 2007 or early, please use the Print (Syntax 2) overloading method.

Note:

For Delphi 3, the method overload isn't supported, so the method names of Syntax 1 and Syntax 2 are changed to Print1 and Print2.

B.1.9.1 Print - Syntax 1

Prints a barcode symbol to printer. The barcode symbol is specified in the properties of this barcode component. Pelase use the method between **Printer.BeginDoc** and **Printer.EndDoc** methods.

Syntax:

```
function Print(Left, Top, Module: Double; BarcodeWidth: Double = 0; BarcodeHeight: Double = 0; Angle: Integer = -1): Integer;
  overload; virtual;
```

Description:

Prints current barcode symbol that is specified in the properties of this barcode component to the printer.

Parameters:

- **Left:** Double; Specifies the margin between the barcode symbol and the left side of the paper in millimeters. If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too.

  See also the "LeftMargin" property.

- **Top:** Double; Specifies the margin between the barcode symbol and the top side of the paper in millimeters. If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too.

  See also the "TopMargin" property.

- **Module:** Double; Specifies the module width in millimeters, it is the width of the smallest bar (or space) in the barcode symbol.

  If the BarcodeWidth parameter is greater than zero, the value in the Module will be ignored, the module value will be calculated based on the BarcodeWidth parameter. If both Module and BarcodeWidth parameters are less than or equal to zero, the BarcodeHeight parameter must be set to greater than zero, the module value will be calculated based on the BarcodeHeight parameter and the Height property.

  See also the "Module" property.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to -1 if the Angle is not provided, and the barcode symbol will be rotated base on the value of the Orientation property:

  - boLeftRight: 0 degrees
  - boRightLeft: 180 degrees
  - boTopBottom: 270 degrees
  - boBottomTop: 90 degrees

  If you want to use the -1 degrees, the 359 degrees can be used instead.

- **BarcodeWidth:** Double, Specifies the barcode symbol width before rotation, in millimeters. If the human readable text is displayed and it exceeds the barcode symbol in horizontal direction, the excess isn't included in the width value.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment property is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), left spacing, and right spacing are included too.

  If the parameter is provided and isn't zero; the value in Module parameter will be ignored, the module width will be calculated based on the BarcodeWidth value. If the parameter isn't provided or it's set to zero, the Module parameter will be used.

  See also the "BarcodeWidth" property.

- **BarcodeHeight:** Integer; Specifies the distance between the top and bottom of the barcode symbol before rotation, in millimeters or modules. If the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess isn't included.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars are included too.

  If the parameter isn't provided or it's set to zero, it will be calculated based on the values of Module parameter and the Height property.

  If the parameter is provided and it is not set to zero, the value of Height property will be ignored. If it's greater than zero, it specifies the height in millimeters. If it's less than zero, the absolute value of the parameter specifies the height in modules.

Return:

- If the method succeeds, the return value is zero.
- If the length of the barcode text that is specified by Barcode or Data (only for Delphi/C++ Builder 2009 or later)property is invalid, the return value is -1. Corresponding to the OnInvalidLength or OnInvalidDataLength (only for Delphi/C++ Builder 2009 or later) event will occur.

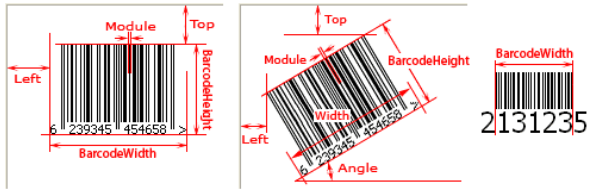
- If all of the Module, BarcodeWidth, and BarcodeHeight parameters are less than or equal to zero, the return value is -2.
- If there is any invalid character in the the barcode text that is specified by Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character. Corresponding to the OnInvalidChar or OnInvalidDataChar (only for Delphi/C++ Builder 2009 or later) event will occur. For the TBarcode1D_OneCode component, if the invalid character in the Tracking property, the index is from 1 to 20 including 1 and 20; if it is in the Routing property, the value starts with 21 (First character of the Routing property).

Note:

Pelase use the method between **Printer.BeginDoc** and **Printer.EndDoc** methods. For example:

```
Printer.BeginDoc;
... { Print other content }
Barcode1D_Code391.Print(...); { Print the barcode }
... { Print other content }
Printer.EndDoc;
```

See diagram:



Note:

For Delphi 3, the method overload and default value of parameter aren't supported, so the method name is changed to Print1, and the parameters Module, BarcodeWidth, BarcodeHeight, and Angle are required.

B.1.9.2 Print - Syntax 2

Prints a barcode symbol to printer. The barcode symbol is specified in the parameters of this method. Pelase use the method between the **Printer.BeginDoc** and the **Printer.EndDoc** methods.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TBarcodeTextDefine = record
    DisplayText: TDisplayText;
    TextPosition: TTextPosition;
    TextAlignment: TTextAlignment;
    TextFont: TFont;
    ExtraFontSize: Integer;
  end;
function Print(Left, Top: Double; Barcode: string; AutoCheckDigit: Boolean; BarColor, SpaceColor: TColor; BarcodeTextDefine:
  TBarcodeTextDefine; Ratio: Double; Module: Double = 0; BarcodeWidth: Integer = 0; BarcodeHeight: Double = 0; Angle: Integer
  = 0): Integer; overload; virtual;
```

Description:

Prints a barcode symbol that is specified in the parameters of this method to the printer.

Parameters:

- **Left:** Double; Specifies the margin between the barcode symbol and the left side of the paper in millimeters. If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too.

  See also the "LeftMargin" property.

- **Top:** Double; Specifies the margin between the barcode symbol and the top side of the paper in millimeters. If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too.

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too.

If the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too.

See also the "TopMargin" property.

- **Barcode:** String; Specifies the barcode text. It is of type string. For Delphi/C++ Builder 2007 or early, theBarcode parameter is in fact an AnsiString. For Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, if you use them in the Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiSting in ANSI encoding scheme. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in theOnEncode event, or specify the converted string in the Barcode paramater. Also, you can use the method to encode a block of binary (bytes) data; If you use them in the Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString. By default, the unicode string will be converted to an ANSI encoding string, then be encoded into the barcode symbol. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in theOnEncode event, or use the Print (Syntax 3) overloading method and specify the converted string in its Data paramater. If you want to encode block of binary (bytes) data, please use the Print (Syntax 3) overloading method. Note, the "\" character is used as a escape prefix, so if you want to encode the "\" character, please use the "\\" instead of it.

  For the TBarcode1D_OneCode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

  For the TBarcode1D_FIM component, it is single character that denotes the FIM type, form "A" to "E", the "E" character dnotes an empty barcode symbol.

  For the TBarcode1D_Patch component, it is single character that denotes the Ptach type, and it can be set to "0", "1", "2", "3", "4", "6", and "T", the "0" character dnotes an empty barcode symbol.

  For the TBarcode1D_Code32 component, First "A" character does not need to be entered in the parameter. Also, for the TBarcode1D_PZN component, First "PZN" characters do not need to be entered in the parameter.

  See also the "Barcode" property.

- **AutoCheckDigit:** Boolean; Specifies whether the check digit should be automatically appended to the barcode symbol.

  See also the "AutoCheckDigit" property.

- **BarColor:** TColor; Specifies the color for all bars in the barcode symbol.

  See also the "BarColor" property.

- **SpaceColor:** TColor; Specifies the color for all spaces in the barcode symbol.

  If the human readable text is represent, the color will be used as its background color (the foreground color is specified using the TextFont field of the BarcodeTextDefine parameter). For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 components, the left spacing and the right spacing will be represented using the color. For TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_UPCA, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones, left quietzone spacing and right quiet zone spacing will be represented using the color. In other words, the parameter specify the background color for entire barcode symbol.

  See also the "SpaceColor" property.

- **BarcodeTextDefine:** TBarcodeTextDefine; Specifies whether to display the human readable text and how to display the human readable text. The record is defined in the pBarcode1D unit.

  See also the TBarcodeTextDefine record.

- **Ratio:** Double; Specifies ratio between a wide bar (or space) and a narrow bar (or space) in the barcode symbol. The normal values are from 2.0 to 3.0. If the parameter is less than or equal to zero, the method fails, and the return value is -2.

  See also the "Ratio" property.

- **Module:** Double; Specifies the module width in millimeters, it is the width of the smallest bar (or space) in the barcode symbol.

  If the BarcodeWidth parameter is greater than zero, the value in the Module will be ignored, the module value will be calculated based on the BarcodeWidth parameter. If both Module and BarcodeWidth parameters are less than or equal to zero, the BarcodeHeight parameter must be set to greater than zero, the module value will be calculated based on the BarcodeHeight parameter and the Height property.

  See also the "Module" property.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to 0 if the Angle is not provided, meaning left to right horizontal direction.

- **BarcodeWidth:** Double, Specifies the barcode symbol width before rotation, in millimeters. If the human readable text is displayed and it exceeds the barcode symbol in horizontal direction, the excess isn't included in the width value.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), left spacing, and right spacing are included too.

  If the parameter is provided and isn't zero; the value in Module parameter will be ignored, the module width will be calculated based on the BarcodeWidth value. If the parameter isn't provided or it's set to zero, the Module parameter will be used.

  See also the "BarcodeWidth" property.

- **BarcodeHeight:** Integer; Specifies the distance between the top and bottom of the barcode symbol before rotation, in millimeters or modules. If the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess isn't included.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars are included too.

If the parameter isn't provided or it's set to zero, it will be calculated based on the values of Module parameter and the Height property.

If the parameter is provided and it is not set to zero, the value of Height property will be ignored. If it's greater than zero, it specifies the height in millimeters. If it's less than zero, the absolute value of the parameter specifies the height in modules.

Return:

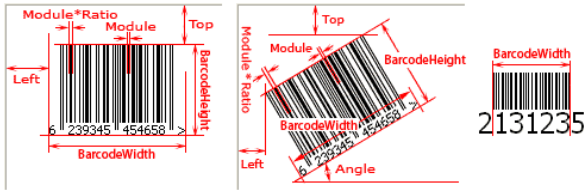
- If the method succeeds, the return value is zero.
- If the string length of Barcode parameter is invalid, the return value is -1.
- If all of the Module, BarcodeWidth, and BarcodeHeight parameters are less than or equal to zero, the return value is -2.
- If the Ratio parameter is less than or equal to zero, the return value is -2.
- If there is any invalid character in the Barcode parameter, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character.

Note:

Pelase use the method between **Printer.BeginDoc** and **Printer.EndDoc** methods. For example:

```
Printer.BeginDoc;
... { Print other content }
Barcode1D_Code391.Print(...); { Print the barcode }
... { Print other content }
Printer.EndDoc;
```

See diagram:



Note:

For Delphi 3, the method overload and default value of parameter aren't supported, so the method name is changed to Print2, and the parameters Module, BarcodeWidth, BarcodeHeight, and Angle are required.

B.1.9.3 Print - Syntax 3

Prints a barcode symbol to printer. The barcode symbol is specified in the parameters of this method. Pelase use the method between the **Printer.BeginDoc** and the **Printer.EndDoc** methods.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TBarcodeTextDefine = record
    DisplayText: TDisplayText;
    TextPosition: TTextPosition;
    TextAlignment: TTextAlignment;
    TextFont: TFont;
    ExtraFontSize: Integer;
  end;
function Print(Left, Top: Double; Data: AnsiString; AutoCheckDigit: Boolean; BarColor, SpaceColor: TColor; BarcodeTextDefine:
  TBarcodeTextDefine; Ratio: Double; Module: Double = 0; BarcodeWidth: Integer = 0; BarcodeHeight: Double = 0; Angle: Integer
  = 0): Integer; overload; virtual;
```

Description:

Prints a barcode symbol that is specified in the parameters of this method to the printer.

Parameters:

- **Left:** Double; Specifies the margin between the barcode symbol and the left side of the paper in millimeters. If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too.

See also the "LeftMargin" property.

- **Top:** Double; Specifies the margin between the barcode symbol and the top side of the paper in millimeters. If the human readable text is represented, it's included in the barcode symbol.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too.

  If the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too.

  See also the "TopMargin" property.

- **Data:** AnsiString; Specifies the barcode text. It is of type AnsiString, so you can specifiy the barcode text in AnsiString format. The method is available only for the Delphi/C++ Builder 2009 or later.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, you can use the method to encode a block of binary (bytes) data under Delphi/C++ Builder 2009 or later. If you want to encode a block of binary (bytes) data into a barcode symbol under Delphi/C++ Builder 2007 or early, please use the Print (Syntax 2) overloading method. Note, the "\" character is used as a escape prefix, so if you want to encode the "\" character, please use the "\\" instead of it.

  For the TBarcode1D_OneCode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

  For the TBarcode1D_FIM component, it is single character that denotes the FIM type, form "A" to "E", the "E" character dnotes an empty barcode symbol.

  For the TBarcode1D_Patch component, it is single character that denotes the Ptach type, and it can be set to "0", "1", "2", "3", "4", "6", and "T", the "0" character dnotes an empty barcode symbol.

  For the TBarcode1D_Code32 component, First "A" character does not need to be entered in the parameter. Also, for the TBarcode1D_PZN component, First "PZN" characters do not need to be entered in the parameter.

  See also the "Data" property.

- **AutoCheckDigit:** Boolean; Specifies whether the check digit should be automatically appended to the barcode symbol.

  See also the "AutoCheckDigit" property.

- **BarColor:** TColor; Specifies the color for all bars in the barcode symbol.

  See also the "BarColor" property.

- **SpaceColor:** TColor; Specifies the color for all spaces in the barcode symbol.

  If the human readable text is represent, the color will be used as its background color (the foreground color is specified using the TextFont field of the BarcodeTextDefine parameter). For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 components, the left spacing and the right spacing will be represented using the color. For TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_UPCA, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the left and right quiet zones, left quietzone spacing and right quiet zone spacing will be represented using the color. In other words, the parameter specify the background color for entire barcode symbol.

  See also the "SpaceColor" property.

- **BarcodeTextDefine:** TBarcodeTextDefine; Specifies whether to display the human readable text and how to display the human readable text. The record is defined in the pBarcode1D unit.

  See also the TBarcodeTextDefine record.

- **Ratio:** Double; Specifies ratio between a wide bar (or space) and a narrow bar (or space) in the barcode symbol. The normal values are from 2.0 to 3.0. If the parameter is less than or equal to zero, the method fails, and the return value is -2.

  See also the "Ratio" property.

- **Module:** Double; Specifies the module width in millimeters, it is the width of the smallest bar (or space) in the barcode symbol.

  If the BarcodeWidth parameter is greater than zero, the value in the Module will be ignored, the module value will be calculated based on the BarcodeWidth parameter. If both Module and BarcodeWidth parameters are less than or equal to zero, the BarcodeHeight parameter must be set to greater than zero, the module value will be calculated based on the BarcodeHeight parameter and the Height property.

  See also the "Module" property.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to 0 if the Angle is not provided, meaning left to right horizontal direction.

- **BarcodeWidth:** Double, Specifies the barcode symbol width before rotation, in millimeters. If the human readable text is displayed and it exceeds the barcode symbol in horizontal direction, the excess isn't included in the width value.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), left spacing, and right spacing are included too.

  If the parameter is provided and isn't zero; the value in Module parameter will be ignored, the module width will be calculated based on the BarcodeWidth value. If the parameter isn't provided or it's set to zero, the Module parameter will be used.

  See also the "BarcodeWidth" property.

- **BarcodeHeight:** Integer; Specifies the distance between the top and bottom of the barcode symbol before rotation, in millimeters or modules. If the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess isn't included.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars are included too.

  If the parameter isn't provided or it's set to zero, it will be calculated based on the values of Module parameter and the Height property.

  If the parameter is provided and it is not set to zero, the value of Height property will be ignored. If it's greater than zero, it specifies the height in millimeters. If it's less than zero, the absolute value of the parameter specifies the height in modules.

Return:

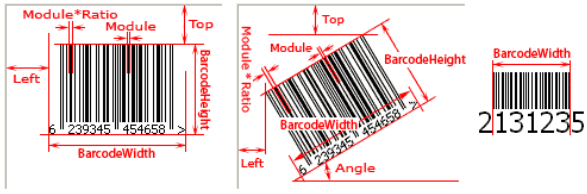
- If the method succeeds, the return value is zero.
- If the string length of Barcode parameter is invalid, the return value is -1.
- If all of the Module, BarcodeWidth, and BarcodeHeight parameters are less than or equal to zero, the return value is -2.
- If the Ratio parameter is less than or equal to zero, the return value is -2.
- If there is any invalid character in the Barcode parameter, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character.

Note:

Pelase use the method between **Printer.BeginDoc** and **Printer.EndDoc** methods. For example:

```
Printer.BeginDoc;
... { Print other content }
Barcode1D_Code391.Print(...); { Print the barcode }
... { Print other content }
Printer.EndDoc;
```

See diagram:



Note:

The overloading method is available only for the Delphi/C++ Builder 2009 or later.

B.1.10 PrintSize

Returns actual horizontal width and vertical height of a rotated barcode symbol in millimeters. There are several different overloading methods, Syntax 1, Syntax 2, and Syntax 3 (only for Delphi/C++Builder 2009 or later):

- **Syntax 1:** Returns the actual print horizontal width and vertical height of the rotated barcode symbol that is specified by the properties of this barcode component.
- **Syntax 2:** Returns the actual print horizontal width and vertical height of the rotated barcode symbol that is specified by the parameters of this method. The barcode text is specified in the Barcode parameter. It is of type string. For Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiString. For Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, if you use them in the Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiSting in ANSI encoding scheme. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or specify the converted string in the Barcode paramater. Also, you can use the method if you encode a block of binary (bytes) data; If you use them in the Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString. By default, the unicode string will be converted to an ANSI encoding string, then be encoded into the barcode symbol. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in the OnEncode event, or use the PrintSize (Syntax 3) overloading method and specify the converted string in its Data paramater. If you encode a block of binary (bytes) data, please use the PrintSize (Syntax 3) overloading method.
- **Syntax 3:** Returns the actual print horizontal width and vertical height of the rotated barcode symbol that is specified by the parameters of this method. The barcode text is specified in the Data parameter. It is of type AnsiString, so you can specifiy the barcode text in AnsiString format. The method is available only for the Delphi/C++ Builder 2009 or later.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, you can use the method if you encode a block of binary (bytes) data under Delphi/C++ Builder 2009 or later. If you encode a block of binary (bytes) data into a barcode symbol under Delphi/C++ Builder 2007 or early, please use the PrintSize (Syntax 2) overloading method.

Note:

For Delphi 3, the method overload isn't supported, so the method names of Syntax 1 and Syntax 2 are changed to PrintSize1 and PrintSize2.

B.1.10.1 PrintSize - Syntax 1

Returns actual size of a rotated barcode symbol in millimeters. The barcode symbol is specified in the properties of this barcode component.

Syntax:

```
function PrintSize(var TotalWidth, TotalHeight, SymbolWidth, SymbolHeight: Double; Module: Double; BarcodeWidth: Double = 0;
  BarcodeHeight: Double = 0; Angle: Integer = -1; HDPI: Integer = 0; VDPI: Integer = 0): Integer; overload; virtual;
```

Description:

The method returns the actual size of the rotated barcode symbol that is specified by properties of this barcode component, in millimeters.

Note, if the DisplayText property isn't set to dtNone, pelase use the method between **Printer.BeginDoc** and **Printer.EndDoc** methods, and the printer must be connected to your computer.

Parameters:

- **TotalWidth:** Double; Returns the horizontal width of the rotated barcode symbol in millimeters.

  Before rotation, if the human readable text is represented, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **TotalHeight:** Double; Returns the vertical height of the rotated barcode symbol in millimeters.

  Before rotation, if the human readable text is represented, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolWidth:** Double; Returns the distance between the leading and trailing of the rotated barcode symbol in millimeters.

  Before rotation, if the human readable text is represented, the width of human readable text will be consulted. Otherwise, it will not be consulted. Note, if the human readable text is represented, and it exceeds the barcode symbol in horizontal direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment property is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolHeight:** Double; Returns the distance between the top and bottom of the rotated barcode symbol in millimeters.

  Before rotation, if the human readable text is represented, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess is included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars (BearerWidth) are included too.

  See also the "Height" property.

- **Module:** Double; Specifies the module width in millimeters, it is the width of the smallest bar (or space) in the barcode symbol.

  If the BarcodeWidth parameter is greater than zero, the value in the Module will be ignored, the module value will be calculated based on the BarcodeWidth parameter. If both Module and BarcodeWidth parameters are less than or equal to zero, the BarcodeHeight parameter must be set to greater than zero, the module value will be calculated based on the BarcodeHeight parameter and the Height property.

  See also the "Module" property.

- **BarcodeWidth:** Double, Specifies the barcode symbol width before rotation, in millimeters. If the human readable text is displayed and it exceeds the barcode symbol in horizontal direction, the excess isn't included in the width value.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment property is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), left spacing, and right spacing are included too.

  If the parameter is provided and isn't zero; the value in Module parameter will be ignored, the module width will be calculated based on the BarcodeWidth value. If the parameter isn't provided or it's set to zero, the Module parameter will be used.

  See also the "BarcodeWidth" property.

- **BarcodeHeight:** Integer; Specifies the distance between the top and bottom of the barcode symbol before rotation, in millimeters or modules. If the human readable text is

displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess isn't included.

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars are included too.

If the parameter isn't provided or it's set to zero, it will be calculated based on the values of Module parameter and the Height property.

If the parameter is provided and it is not set to zero, the value of Height property will be ignored. If it's greater than zero, it specifies the height in millimeters. If it's less than zero, the absolute value of the parameter specifies the height in modules.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to -1 if the Angle is not provided, and the barcode symbol will be rotated base on the value of the Orientation property:
  - boLeftRight: 0 degrees
  - boRightLeft: 180 degrees
  - boTopBottom: 270 degrees
  - boBottomTop: 90 degrees

  If you want to use the -1 degrees, the 359 degrees can be used instead.
- **HDPI:** Integer, Specifies the physical horizontal resolution of printer in DPI (dots per inch).

  It defaults to 0 if the HDPI is not provided. If it is set to less than or equal to zero, the horizontal resolution will be obtained from your printer.
- **VDPI:** Integer, Specifies the physical vertical resolution of printer in DPI (dots per inch).

  It defaults to 0 if the VDPI is not provided. If it is set to less than or equal to zero, the vertical resolution will be obtained from your printer.

Return:

- If the method succeeds, the return value is zero.
- If the length of the barcode text that is specified by Barcode or Data (only for Delphi/C++ Builder 2009 or later)property is invalid, the return value is -1. Corresponding to the OnInvalidLength or OnInvalidDataLength (only for Delphi/C++ Builder 2009 or later) event will occur.
- If all of the Module, BarcodeWidth, and BarcodeHeight parameters are less than or equal to zero, the return value is -2.
- If there is any invalid character in the the barcode text that is specified by Barcode or Data (only for Delphi/C++ Builder 2009 or later) property, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character. Corresponding to the OnInvalidChar or OnInvalidDataChar (only for Delphi/C++ Builder 2009 or later) event will occur. For the TBarcode1D_OneCode component, if the invalid character in the Tracking property, the index is from 1 to 20 including 1 and 20; if it is in the Routing property, the value starts with 21 (First character of the Routing property).

See diagram:

Note:

For Delphi 3, the method overload and default value of parameter aren't supported, so the method name is changed to PrintSize1, and the parameters Module, BarcodeWidth, BarcodeHeight, Angle, HDPI, and VDPI are required.

B.1.10.2 PrintSize - Syntax 2

Returns actual size of a rotated barcode symbol in millimeters. The barcode symbol is specified in the parameters of this method.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TBarcodeTextDefine = record
    DisplayText: TDisplayText;
    TextPosition: TTextPosition;
    TextAlignment: TTextAlignment;
    TextFont: TFont;
    ExtraFontSize: Integer;
  end;

function PrintSize(var TotalWidth, TotalHeight, SymbolWidth, SymbolHeight: Double; Barcode: string; AutoCheckDigit: Boolean;
  BarcodeTextDefine: TBarcodeTextDefine; Ratio: Double; Module: Double = 0; BarcodeWidth: Double = 0; BarcodeHeight: Double =
  0; Angle: Integer = 0; HDPI: Integer = 0; VDPI: Integer = 0): Integer; overload; virtual;
```

Description:

The method returns the actual size of a rotated barcode symbol that is specified by parameters of this method, in millimeter.

Note, if the Display field of BarcodeTextDefine parameter isn't set to dtNone, pelase use the method between **Printer.BeginDoc** and **Printer.EndDoc** methods, and the printer must be connected to your computer.

Parameters:

- **TotalWidth:** Double; Returns the horizontal width of the rotated barcode symbol in millimeters.

  Before rotation, if the human readable text is represented, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **TotalHeight:** Double; Returns the vertical height of the rotated barcode symbol in millimeters.

  Before rotation, if the human readable text is represented, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolWidth:** Double; Returns the distance between the leading and trailing of the rotated barcode symbol in millimeters.

  Before rotation, if the human readable text is represented, the width of human readable text will be consulted. Otherwise, it will not be consulted. Note, if the human readable text is represented, and it exceeds the barcode symbol in horizontal direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolHeight:** Double; Returns the distance between the top and bottom of the rotated barcode symbol in millimeters.

  Before rotation, if the human readable text is represented, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess is included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars (BearerWidth) are included too.

  See also the "Height" property.

- **Barcode:** String; Specifies the barcode text. It is of type string. For Delphi/C++ Builder 2007 or early, theBarcode parameter is in fact an AnsiString. For Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, if you use them in the Delphi/C++ Builder 2007 or early, the Barcode parameter is in fact an AnsiSting in ANSI encoding scheme. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in theOnEncode event, or specify the converted string in the Barcode paramater. Also, you can use the method if you encode a block of binary (bytes) data; If you use them in the Delphi/C++ Builder 2009 or later, it is in fact an UnicodeString instead of AnsiString. By default, the unicode string will be converted to an ANSI encoding string, then be encoded into the barcode symbol. If you want to use other encoding scheme (for example the UTF-8, UTF-16), please convert it in theOnEncode event, or use the PrintSize (Syntax 3)overloading method and specify the converted string in its Data paramater. If you encode a block of binary (bytes) data, please use the PrintSize (Syntax 3) overloading method. Note, the "\" character is used as a escape prefix, so if you want to encode the "\" character, please use the "\\" instead of it.

  For the TBarcode1D_OneCode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

  For the TBarcode1D_FIM component, it is single character that denotes the FIM type, form "A" to "E", the "E" character dnotes an empty barcode symbol.

For the TBarcode1D_Patch component, it is single character that denotes the Ptach type, and it can be set to "0", "1", "2", "3", "4", "6", and "T", the "0" character dnotes an empty barcode symbol.

For the TBarcode1D_Code32 component, First "A" character does not need to be entered in the parameter. Also, for the TBarcode1D_PZN component, First "PZN" characters do not need to be entered in the parameter.

See also the "Barcode" property.

- **AutoCheckDigit:** Boolean; Specifies whether the check digit should be automatically appended to the barcode symbol.

  See also the "AutoCheckDigit" property.

- **BarcodeTextDefine:** TBarcodeTextDefine; Specifies whether to display the human readable text and how to display the human readable text. The record is defined in the pBarcode1D unit.

  See also the TBarcodeTextDefine record.

- **Ratio:** Double; Specifies ratio between a wide bar (or space) and a narrow bar (or space) in the barcode symbol. The normal values are from 2.0 to 3.0. If the parameter is less than or equal to zero, the method fails, and the return value is -2.

  See also the "Ratio" property.

- **Module:** Double; Specifies the module width in millimeters, it is the width of the smallest bar (or space) in the barcode symbol.

  If the BarcodeWidth parameter is greater than zero, the value in the Module will be ignored, the module value will be calculated based on the BarcodeWidth parameter. If both Module and BarcodeWidth parameters are less than or equal to zero, the BarcodeHeight parameter must be set to greater than zero, the module value will be calculated based on the BarcodeHeight parameter and the Height property.

  See also the "Module" property.

- **BarcodeWidth:** Double, Specifies the barcode symbol widthbefore rotation, in millimeters. If the human readable text is displayed and it exceeds the barcode symbol in horizontal direction, the excess isn't included in the width value.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), left spacing, and right spacing are included too.

  If the parameter is provided and isn't zero; the value in Module parameter will be ignored, the module width will be calculated based on the BarcodeWidth value. If the parameter isn't provided or it's set to zero, the Module parameter will be used.

  See also the "BarcodeWidth" property.

- **BarcodeHeight:** Integer; Specifies the distance between the top and bottom of the barcode symbol before rotation, in millimeters or modules. If the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess isn't included.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars are included too.

  If the parameter isn't provided or it's set to zero, it will be calculated based on the values of Module parameter and the Height property.

  If the parameter is provided and it is not set to zero, the value of Height property will be ignored. If it's greater than zero, it specifies the height in millimeters. If it's less than zero, the absolute value of the parameter specifies the height in modules.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to 0 if the Angle is not provided, meaning left to right horizontal direction.

- **HDPI:** Integer, Specifies the physical horizontal resolution of printer in DPI (dots per inch).

  It defaults to 0 if the HDPI is not provided. If it is set to less than or equal to zero, the horizontal resolution will be obtained from your printer.

- **VDPI:** Integer, Specifies the physical vertical resolution of printer in DPI (dots per inch).

  It defaults to 0 if the VDPI is not provided. If it is set to less than or equal to zero, the vertical resolution will be obtained from your printer.

Return:

- If the method succeeds, the return value is zero.
- If the string length of Barcode parameter is invalid, the return value is -1.
- If all of the Module, BarcodeWidth, and BarcodeHeight parameters are less than or equal to zero, the return value is -2.
- If the Ratio parameter is less than or equal to zero, the return value is -2.
- If there is any invalid character in the Barcode parameter, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character.

See diagram:

Note:

For Delphi 3, the method overload and default value of parameter aren't supported, so the method name is changed to PrintSize2, and the parameters Module, BarcodeWidth, BarcodeHeight, Angle, HDPI, and VDPI are required.

B.1.10.3 PrintSize - Syntax 3

Returns actual size of a rotated barcode symbol in millimeters. The barcode symbol is specified in the parameters of this method.

Syntax:

```
type
  { Defined in the pCore1D unit }
  TBarcodeTextDefine = record
    DisplayText: TDisplayText;
    TextPosition: TTextPosition;
    TextAlignment: TTextAlignment;
    TextFont: TFont;
    ExtraFontSize: Integer;
  end;

function PrintSize(var TotalWidth, TotalHeight, SymbolWidth, SymbolHeight: Double; Data: AnsiString; AutoCheckDigit: Boolean;
  BarcodeTextDefine: TBarcodeTextDefine; Ratio: Double; Module: Double = 0; BarcodeWidth: Double = 0; BarcodeHeight: Double =
  0; Angle: Integer = 0; HDPI: Integer = 0; VDPI: Integer = 0): Integer; overload; virtual;
```

Description:

The method returns the actual size of a rotated barcode symbol that is specified by parameters of this method, in millimeter.

Note, if the Display field of BarcodeTextDefine parameter isn't set to dtNone, pelase use the method between **Printer.BeginDoc** and **Printer.EndDoc** methods, and the printer must be connected to your computer.

Parameters:

- **TotalWidth:** Double; Returns the horizontal width of the rotated barcode symbol in millimeters.

  Before rotation, if the human readable text is represented, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **TotalHeight:** Double; Returns the vertical height of the rotated barcode symbol in millimeters.

  Before rotation, if the human readable text is represented, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is represented and it exceeds the barcode symbol in horizontal or vertical direction, the excess is

included.

For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolWidth:** Double; Returns the distance between the leading and trailing of the rotated barcode symbol in millimeters.

  Before rotation, if the human readable text is represented, the width of human readable text will be consulted. Otherwise, it will not be consulted. Note, if the human readable text is represented, and it exceeds the barcode symbol in horizontal direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is represented, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolHeight:** Double; Returns the distance between the top and bottom of the rotated barcode symbol in millimeters.

  Before rotation, if the human readable text is represented, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess is included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars (BearerWidth) are included too.

  See also the "Height" property.

- **Data:** AnsiString; Specifies the barcode text. It is of type AnsiString, so you can specifiy the barcode text in AnsiString format. The method is available only for the Delphi/C++ Builder 2009 or later.

  For the TBarcode1D_Code128 and the TBarcode1D_EAN128 components, you can use the method if you encode a block of binary (bytes) data under Delphi/C++ Builder 2009 or later. If you encode a block of binary (bytes) data into a barcode symbol under Delphi/C++ Builder 2007 or early, please use the PrintSize (Syntax 2) overloading method. Note, the "\" character is used as a escape prefix, so if you want to encode the "\" character, please use the "\\" instead of it.

  For the TBarcode1D_OneCode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

  For the TBarcode1D_FIM component, it is single character that denotes the FIM type, form "A" to "E", the "E" character dnotes an empty barcode symbol.

  For the TBarcode1D_Patch component, it is single character that denotes the Ptach type, and it can be set to "0", "1", "2", "3", "4", "6", and "T", the "0" character dnotes an empty barcode symbol.

  For the TBarcode1D_Code32 component, First "A" character does not need to be entered in the parameter. Also, for the TBarcode1D_PZN component, First "PZN" characters do not need to be entered in the parameter.

  See also the "Data" property.

- **AutoCheckDigit:** Boolean; Specifies whether the check digit should be automatically appended to the barcode symbol.

  See also the "AutoCheckDigit" property.

- **BarcodeTextDefine:** TBarcodeTextDefine; Specifies whether to display the human readable text and how to display the human readable text. The record is defined in the pBarcode1D unit.

  See also the TBarcodeTextDefine record.

- **Ratio:** Double; Specifies ratio between a wide bar (or space) and a narrow bar (or space) in the barcode symbol. The normal values are from 2.0 to 3.0. If the parameter is less than or equal to zero, the method fails, and the return value is -2.

  See also the "Ratio" property.

- **Module:** Double; Specifies the module width in millimeters, it is the width of the smallest bar (or space) in the barcode symbol.

  If the BarcodeWidth parameter is greater than zero, the value in the Module will be ignored, the module value will be calculated based on the BarcodeWidth parameter. If both Module and BarcodeWidth parameters are less than or equal to zero, the BarcodeHeight parameter must be set to greater than zero, the module value will be calculated based on the BarcodeHeight parameter and the Height property.

  See also the "Module" property.

- **BarcodeWidth:** Double, Specifies the barcode symbol widthbefore rotation, in millimeters. If the human readable text is displayed and it exceeds the barcode symbol in horizontal direction, the excess isn't included in the width value.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment field of the BarcodeTextDefine parameter is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, width of left and right bearer bars (BearerWidth), left spacing, and right spacing are included too.

  If the parameter is provided and isn't zero; the value in Module parameter will be ignored, the module width will be calculated based on the BarcodeWidth value. If the parameter isn't provided or it's set to zero, the Module parameter will be used.

  See also the "BarcodeWidth" property.

- **BarcodeHeight:** Integer; Specifies the distance between the top and bottom of the barcode symbol before rotation, in millimeters or modules. If the human readable text is

displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess isn't included.

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars are included too.

If the parameter isn't provided or it's set to zero, it will be calculated based on the values of Module parameter and the Height property.

If the parameter is provided and it is not set to zero, the value of Height property will be ignored. If it's greater than zero, it specifies the height in millimeters. If it's less than zero, the absolute value of the parameter specifies the height in modules.

- **Angle:** Integer; Specifies an angle in degrees to rotate the barcode symbol. It defaults to 0 if the Angle is not provided, meaning left to right horizontal direction.
- **HDPI:** Integer, Specifies the physical horizontal resolution of printer in DPI (dots per inch).

  It defaults to 0 if the HDPI is not provided. If it is set to less than or equal to zero, the horizontal resolution will be obtained from your printer.
- **VDPI:** Integer, Specifies the physical vertical resolution of printer in DPI (dots per inch).

  It defaults to 0 if the VDPI is not provided. If it is set to less than or equal to zero, the vertical resolution will be obtained from your printer.

Return:

- If the method succeeds, the return value is zero.
- If the string length of Barcode parameter is invalid, the return value is -1.
- If all of the Module, BarcodeWidth, and BarcodeHeight parameters are less than or equal to zero, the return value is -2.
- If the Ratio parameter is less than or equal to zero, the return value is -2.
- If there is any invalid character in the Barcode parameter, the return value is the position index of first invalid character, the index 1 denotes that the first character is invalid character.

See diagram:

Note:

The overloading method is available only for the Delphi/C++ Builder 2009 or later.

B.1.11 Size

Returns the horizontal width and vertical height of a rotated barcode symbol that's displyed in the TImage, TQRImage, or TQRGzImage control specified in the Image property, in pixels.

Syntax:

```
function Size(var Width, Height, SymbolWidth, SymbolHeight: Integer): Boolean; virtual;
```

Description:

The method returns the horizontal width and vertical height of the rotated barcode symbol that is displyed in the TImage, TQRImage, or TQRGzImage control specified in the Image property, in pixels.

Parameters:

- **Width:** Integer; Returns the horizontal width of the rotated barcode symbol that's displyed in the TImage, TQRImage, or TQRGzImage control specified in the Image property, in pixels.

  Before rotation, if the human readable text is displayed, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be

consulted. Otherwise, they will not be consulted. Note, if the human readable text is displayed and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment property is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **Height:** Integer; Returns the vertical height of the rotated barcode symbol that's displyed in the TImage, TQRImage, or TQRGzImage control specified in the Image property, in pixels.

  Before rotation, if the human readable text is displayed, both the width of human readable text and the height of the human readable text (including its vertical spacing) will be consulted. Otherwise, they will not be consulted. Note, if the human readable text is displayed and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment property is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the height of top and bottom bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolWidth:** Integer; Returns the distance between the leading and trailing of the rotated barcode symbol that's displyed in the TImage, TQRImage, or TQRGzImage control specified in the Image property, in pixels.

  Before rotation, if the human readable text is displayed, the width of human readable text will be consulted. Otherwise, it will not be consulted. Note, if the human readable text is displayed, and it exceeds the barcode symbol in horizontal direction, the excess is included.

  For TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment property is set to taCustom, the width of quiet zone marks and their horizontal spacing (LeftQuietZoneSpacing and RightQuietZoneSpacing) are included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the width of left and right bearer bars (BearerWidth), the left spacing, and the right spacing are included too.

- **SymbolHeight:** Integer; Returns the distance between the top and bottom of the rotated barcode symbol that's displyed in the TImage, TQRImage, or TQRGzImage control specified in the Image property, in pixels.

  Before rotation, if the human readable text is displayed, the height of the human readable text and its vertical spacing (TextVSpacing) are included. If it exceeds the barcode symbol in vertical direction, the excess is included too.

  For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, the height of top and bottom bearer bars (BearerWidth) are included too.

  See also the "Height" property.

Return:

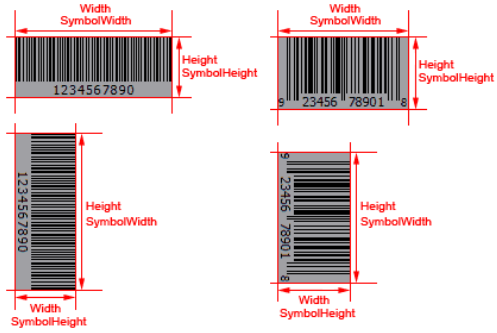
- If the method succeeds, the return value is true.
- If the length of Barcode or Data (only for Delphi/C++ Builder 2009 or later) property value is invalid, the return value is false, corresponding to the OnInvalidLength or OnInvalidDataLength (only for Delphi/C++ Builder 2009 or later) event will occur.

  If there is any invalid character in the Barcode or Data (only for Delphi/C++ Builder 2009 or later) property value, the return value is false, corresponding to the OnInvalidChar or OnInvalidDataChar (only for Delphi/C++ Builder 2009 or later) event will occur.

Note:

If the Image property is not set, the human readable text will not be consulted.

See diagram:



Annex C. Events

C.1 TBarcode1D

C.1.1 OnChange

Occurs when the barcode symbol of the barcode component is changed, include following properties:

- Barcode (If exists)
- Data (Only for Delphi/C++ Builder 2009 or later)
- FIMType (Only for TBarcode1D_FIM)
- Tracking (Only for TBarcode1D_OneCode)
- Routing (Only for TBarcode1D_OneCode)
- AutoCheckDigit (If exists)
- Padding (Only for TBarcode1D_Code25Interleaved)
- StartCode (Only for TBarcode1D_Codabar)
- StopCode (Only for TBarcode1D_Codabar)
- NumberCheckDigit (Only for TBarcode1D_Code11)
- CheckMethod (Only for TBarcode1D_MSI)
- Mod11Weighting (Only for TBarcode1D_MSI)
- ExtraChar (Only for TBarcode1D_Telepen)
- OddEncode (Only for TBarcode1D_Telepen)
- Bidirectional (Only for TBarcode1D_Plessey)
- UKMode (Only for TBarcode1D_Plessey)
- CheckStart (Only for TBarcode1D_EAN128)
- CheckLength (Only for TBarcode1D_EAN128)
- Link2D (Only for TBarcode1D_EAN128)

Syntax:

```
property OnChange: TNotifyEvent;
```

Parameters:

- **Sender:** TObject; It is the object whose event handler is called.

C.1.2 OnDecodeText

(TBarcode1D_Code128, TBarcode1D_EAN128)

For the TBarcode1D_Code128 and TBarcode1D_EAB128 components, you can encode a block of binary (bytes) data into the barcode symbol. The event is useful to decode the text from the block of binary (bytes) data in order to output it as the barcode text into the barcode symbol.

For the Delphi/C++ Builder 2007 or early, it occurs when the value ofBarcode property is changed and the component is updating its barcode symbol, or one of the Clear, Draw, Size, DrawTo (Syntax 1 and 2), DrawToSize (Syntax 1 and 2), Print (Syntax 1 and 2), PrintSize (Syntax 1 and 2)method is called. Note, the event function is required only when you encode a block of binary (bytes) data into the barcode symbol.

For the Delphi/C++ Builder 2009 or later, if you use theData property to encode a block of binary (bytes) data into the barcode symbol, it occurs when the value of Data property is changed and the component is updating its barcode symbol, or one of the Clear, Draw, Size, DrawTo (Syntax 1), DrawToSize (Syntax 1), Print (Syntax 1), PrintSize (Syntax 1)method is called. Also, it occurs when one of the DrawTo (Syntax 3), DrawToSize (Syntax 3), Print (Syntax 3), PrintSize (Syntax 3) method is called.

Syntax:

```
type
  { Defined in the pCode128 unit }
  TOnDecodeText = procedure (Sender: TObject; var BarcodeText: string; Data: AnsiString); of object;
property OnDecodeText: TOnDecodeText;
```

Parameters:

- **Sender**: TObject; It is the object whose event handler is called.
- **BarcodeText**: String; The text that's decoded from the block of binary (bytes) data should be returned in the parameter. It will be output as the barcode text in the barcode symbol if

the DisplayText property is not set to dtNone. By default, the block of binary (bytes) data will be as the barcode text directly. For the Delphi/C++ Builder 2009 or later, it will be converted to the UNICODE string firstly.

- **Data**: AnsiString; Its is the block of binary (bytes) data.

  For Delphi/C++ Bilder 2007 or early. If the Barcode property is changed, or one of the Clear, Draw, Size, DrawTo (Syntax 1), DrawToSize (Syntax 1), Print (Syntax 1), PrintSize (Syntax 1) method is called, it's equal to the value of the Barcode property. If one of the DrawTo (Syntax 2), DrawToSize (Syntax 2), Print (Syntax 2), PrintSize (Syntax 2) method is called, it's equal to the value of its Barcode parameter.

  For Delphi/C++ Bilder 2009 or later. If the Data property is changed, or one of the Clear, Draw, Size, DrawTo (Syntax 1), DrawToSize (Syntax 1), Print (Syntax 1), PrintSize (Syntax 1) method is called, it's equal to the value of the Data property. If one of the DrawTo (Syntax 3), DrawToSize (Syntax 3), Print (Syntax 3), PrintSize (Syntax 3) method is called, it's equal to the value of its Data parameter.

  Note: If the AutoCheckDigit property is set to true, and the DisplayText property is set to dtFullEncoded, the check digit will be inserted to the parameter too.

Note:

For the Delphi/C++ Builder 2009 or later, the event doesn't occur when the DrawTo (syntax 2), DrawToSize (syntax 2), Print (syntax 2), or PrintSize (syntax 2) method is called.

C.1.3 OnDrawBarcode

Occurs after representing the barcode symbol. Write an OnDrawBarcode event handler to modify the barcode symbol after it was represented.

Syntax:

```
type
  { Defined in the pBarcode1D unit }
  TBarcodeCustomParameters = record
    Alpha: Double;
    Origin: TPoint;
    Offset: TPoint;
    DensityRate: Double;
    FullEncoded: string;
    Text: string;
    DisplayText: TDisplayText;
    TextPosition: TTextPosition;
    TextAlignment: TTextAlignment;
    TextFont: TFont;
    ExtraFontSize: Integer;
    LeftQuietZone_Spacing: Integer;
    RightQuietZone_Spacing: Integer;
    LeftQuietZone_Width: Integer;
    RightQuietZone_Width: Integer;
    LeftQuietText_Height: Integer;
    RightQuietText_Height: Integer;
    Symbol_Width: Integer;
    Symbol_Height: Integer;
    Symbol_V_Offset: Integer;
    Symbol_H_Offset: Integer;
    Total_Left: Integer;
    Total_Top: Integer;
    Total_Width: Integer;
    Total_Height: Integer;
  end;
  { Defined in the pBarcode1D unit }
  TOnDrawBarcode = procedure (Sender: TObject; Canvas: TCanvas; Parameters: TBarcodeCustomParameters) of object;
property OnDrawBarcode: TOnDrawBarcode;
```

Parameters:

- **Sender:** TObject; It is the object whose event handler is called.
- **Canvas:** TCanvas; The target canvas, the barcode symbol will be represented in it.
- **BarcodeParameters:** TBarcodeCustomParameters; It contains fields to specify the parameters (e.g. position, size, etc.) for the barcode symbol. It is defined in the pBarcode1D unit. See also the TBarcodeCustomParameters record.

C.1.4 OnDrawText

Occurs at discrete points during the representing (either displaying or printing) of the human readable text. Write an OnDrawText event handler to customize the representing of the human readable text at various stages before it is represented.

Syntax:

```
type
  { Defined in the pBarcode1D unit }
  TBarcodeCustomParameters = record
    Alpha: Double;
    Origin: TPoint;
    Offset: TPoint;
    DensityRate: Double;
    FullEncoded: string;
    Text: string;
    DisplayText: TDisplayText;
    TextPosition: TTextPosition;
    TextAlignment: TTextAlignment;
    TextFont: TFont;
    ExtraFontSize: Integer;
    LeftQuietZone_Spacing: Integer;
    RightQuietZone_Spacing: Integer;
    LeftQuietZone_Width: Integer;
    RightQuietZone_Width: Integer;
    LeftQuietText_Height: Integer;
    RightQuietText_Height: Integer;
    Symbol_Width: Integer;
    Symbol_Height: Integer;
    Symbol_V_Offset: Integer;
    Symbol_H_Offset: Integer;
    Total_Left: Integer;
    Total_Top: Integer;
    Total_Width: Integer;
    Total_Height: Integer;
  end;
  { Defined in the pCore1D unit }
  TDrawTextStep = (dtsPrepare, dtsClean, dtsPrint);
  { Defined in the pBarcode1D unit }
  TOnDrawText = procedure (Sender: TObject; Canvas: TCanvas; Step: TDrawTextStep; var BarcodeParameters:
    TBarcodeCustomParameters; var Continue: Boolean; var LeftTop, RightBottom: TPoint; var PDx: PInteger) of object;
property OnDrawText: TOnDrawText;
```

Parameters:

- **Sender:** TObject; It is the object whose event handler is called.
- **Canvas:** TCanvas; The target canvas, the human readable text will be represented in it.
- **Step:** TDrawTextStep; Determine when an OnDrawText event occurs, It can have one of following values (defined in the pCore1D unit):
  - **dtsPrepare:** Occurs before calculate the position and size of the human readable text. You can customizate the human readable text in this step.
  - **dtsClean:** Occurs before erasing the background for human readable text.
  - **dtsPrint:** Occurs before drawing(printing) the human readable text.
- **BarcodeParameters:** TBarcodeCustomParameters; It contains fields to specify the parameters (e.g. position, size, etc.) for the barcode symbol. It is defined in the pBarcode1D unit. If the Step parameter is dtsPrepare, you can change the values of Text, TextFont, TextPosition and TextAlignment fields, all changes to other fields will be ignored. The human readable text will not be displayed if you change the Text field to empty string. If the Step parameter is dtsClear or dtsPrint, all changes to the parameter will be ignored. The Orgin field is only available when Step parameter is dtsClear or dtsPrint. For the values of Symbol_V_Offset, Symbol_H_Offset, Total_Width and Total_Height fields, the human readable text aren't consulted if the Step parameter is dtsPrepare. See also the TBarcodeCustomParameters record.
- **Continue:** Boolean; Indicates whether the human readable text should continue with the default painting after the event handler exits. Set Continue to false to prevent the drawing of the human readable text after event handler exits. If Continue remains set to true, the human readable text continues with the default painting process.
- **LeftTop:** TPoint; The Point of the upper-left corner of the human readable text area before the barcode symbol is rotated, in pixels (Draw, DrawTo) or dots (Print). It is only available when Step parameter is dtsClear or dtsPrint. The point (0, 0) refers to the upper-left corner of entire barcode symbol (If the human readable text is displayed, it's included in the barcode symbol. For the TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment property or the ATextAlignment parameter is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too. For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too. If the human readable text is displayed and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too).
- **RightBottom:** TPoint; The Point of the lower-right corner of the human readable text area before the barcode symbol is rotated, in pixels (Draw, DrawTo) or dots (Print). It is only available when Step parameter is dtsClear or dtsPrint. The point (0, 0) refers to the upper-left corner of entire barcode symbol (If the human readable text is displayed, it's included in the barcode symbol. For the TBarcode1D_UPCA, TBarcode1D_UPCE, TBarcode1D_UPCE0, TBarcode1D_UPCE1, TBarcode1D_EAN2, TBarcode1D_EAN5, TBarcode1D_EAN8, and TBarcode1D_EAN13 barcode components, if the human readable text is displayed, and the TextAlignment property or the ATextAlignment parameter is set to taCustom, the left and right quiet zones marks and their horizontal spacing are included in the barcode symbol too. For TBarcode1D_ITF6, TBarcode1D_ITF14, and TBarcode1D_ITF16 barcode components, all bearer bars, the left and right spacing are included in the barcode symbol too. If the human readable text is displayed and it exceeds the barcode symbol in horizontal or vertical direction, the excess is included in the barcode symbol too).
- **PDx:** PInteger; Points to first element of an array, the array values indicate the distance between origins of adjacent character cells. It is only available when Step parameter is dtsPrint.

See diagram:

C.1.5 OnEncode

(TBarcode1D_Code128, TBarcode1D_EAN128)

For the TBarcode1D_Code128 and TBarcode1D_EAB128 components, occurs when the value of Barcode property is changed and the component is updating its barcode symbol, or one of the Clear, Draw, Size, DrawTo, DrawToSize, Print, PrintSize method is called.

The TBarcode1D_Code128 and TBarcode1D_EAB128 components can be used to encode any character, by default, the ANSI encoding scheme is used. The event is useful if you want to encode the barcode text in your own encoding scheme before generate the barcode symbol. See also the "How to encode the UNICODE text in a Code128/EAN128 symbol" article.

Syntax:

```
type
  { Defined in the pCode128 unit }
  TOnEncode = procedure (Sender: TObject; var Data: AnsiString; Barcode: string) of object;
property OnEncode: TOnEncode;
```

Parameters:

- **Sender**: TObject; It is the object whose event handler is called.
- **Data**: AnsiString; The barcode text that's encoded by your own encoding scheme should be returned in this parameter. Note, the "\" character is used as a escape prefix, so if there is any "\" character in the encoded string except the escape squences, please use the "\\" instead of it.

  The initial value is the barcode text in AnsiString (the barcode text will be converted to AnsiSring if the UNICODE string is supported by the Delphi or C++ Builder). By default, it's in ANSI encoding scheme.
- **Barcode**: String; It is the barcode text in string type (it's an UnicodeString if the UNICODE string is supported by the Delphi or C++ Builder, otherwise it's an AnsiString).

  Its value is equal to the Barcode property if the Clear, Draw, Size, DrawTo (syntax 1), DrawToSize (syntax 1), Print (syntax 1), or PrintSize (syntax 1) method is called or a component is updating its barcode symbol. And the value of the parameter is equal to the Barcode parameter if the DrawTo (syntax 2), DrawToSize (syntax 2), Print (syntax 2), or PrintSize (syntax 2) method is called.

Note:

The event doesn't occur when the DrawTo (syntax 3), DrawToSize (syntax 3), Print (syntax 3), or PrintSize (syntax 3) method is called.

C.1.6 OnInvalidChar

Occurs if there is any invalid character in the Barcode, FIMType (Only for TBarcode1D_FIM component), PatchType (Only for TBarcode1D_Patch component), Tracking (Only for TBarcode1D_OneCode component), or Routing (Only for TBarcode1D_OneCode component) property.

Syntax:

```
type
  { Defined in the pBarcode1D unit }
  TOnInvalidChar = procedure (Sender: TObject; Index: Integer; BarcodeChar: Char) of object;
property OnInvalidChar: TOnInvalidChar;
```

Parameters:

- **Sender:** TObject; It is the object whose event handler is called.
- **Index:** Integer; The index position of first invalid character in the Barcode, FIMType (Only for TBarcode1D_FIM component), PatchType (Only for TBarcode1D_Patch component), Tracking (Only for TBarcode1D_OneCode component), or Routing (Only for TBarcode1D_OneCode component) property, the index 1 denotes that the first character is invalid character.

  For the TBarcode1D_OneCode component, if the invalid character is in the Tracking property, the Index is from 1 to 20 including 1 and 20; if it is in the the Routing property, the value starts with 21 (First character of the Routing property).
- **BarcodeChar:** Char; The first invalid character in the barcode text that is specified by the Barcode, FIMType (Only for TBarcode1D_FIM component), PatchType (Only for TBarcode1D_Patch component), Tracking (Only for TBarcode1D_OneCode component), or Routing (Only for TBarcode1D_OneCode component) property. The character is a WideChar if the UNICODE is supported by the Delphi or C++ Builder. Otherwise it is an AnsiChar.

Note:

- If the Locked property is set to false, the event occurs when any component property is changed to cause the barcode is redrew, or when the Draw, Clear, Size, DrawTo (Syntax 1), DrawToSize (Syntax 1), Print (Syntax 1), or PrintSize (Syntax 1) method is called. Even if the Image property isn't specified.
- If the Locked property is set to true, the event occurs when the Locked property is set to false to cause the barcode is redrew, or the Draw, Clear, Size, DrawTo (Syntax 1), DrawToSize (Syntax 1), Print (Syntax 1), or PrintSize (Syntax 1) method is called. Even if the Image property isn't specified.

C.1.7 OnInvalidDataChar

If you use the Delphi/C++ Builder 2009 or later, you can use the Data property to specify a barcode text in AnsiString format, and encode it into the barcode symbol. Or use the Data property to encode a block of binary (bytes) data into the barcode symbol (only for the TBarcode1D_Code128 and the TBarcode1D_EAN128 components). The event occurs if there is any invalid character in the Data property.

Syntax:

```
type
  { Defined in the pBarcode1D unit }
  TOnInvalidDataChar = procedure (Sender: TObject; Index: Integer; DataChar: AnsiChar) of object;
property OnInvalidDataChar: TOnInvalidDataChar;
```

Parameters:

- **Sender:** TObject; It is the object whose event handler is called.
- **Index:** Integer; The index position of first invalid character in the Data property, the index 1 denotes that the first character is invalid character.
- **DataChar:** AnsiChar; The first invalid character in the barcode text that is specified by the Data property.

Note:

- If the Locked property is set to false, the event occurs when any component property is changed to cause the barcode is redrew, or when the Draw, Clear, Size, DrawTo (Syntax 1), DrawToSize (Syntax 1), Print (Syntax 1), or PrintSize (Syntax 1) method is called. Even if the Image property isn't specified.
- If the Locked property is set to true, the event occurs when the Locked property is set to false to cause the barcode is redrew, or the Draw, Clear, Size, DrawTo (Syntax 1), DrawToSize (Syntax 1), Print (Syntax 1), or PrintSize (Syntax 1) method is called. Even if the Image property isn't specified.
- The event is available only for the Delphi/C++ Builder 2009 or later.

C.1.8 OnInvalidDataLength

If you use the Delphi/C++ Builder 2009 or later, you can use the Data property to specify a barcode text in AnsiString format, and encode it into the barcode symbol. Or use the Data property to encode a block of binary (bytes) data into the barcode symbol (only for the TBarcode1D_Code128 and the TBarcode1D_EAN128 components). The event occurs when the length of the Data property is invalid.

Syntax:

```
type
  { Defined in the pBarcode1D unit }
  TOnInvalidDataLength = procedure (Sender: TObject; Data: AnsiString) of object;
property OnInvalidDataLength: TOnInvalidDataLength;
```

Parameters:

- **Sender:** TObject; It is the object whose event handler is called.
- **Data:** AnsiString; The invalid value of the Data property.

Note:

- If the Locked property is set to false, the event occurs when any component property is changed to cause the barcode is redrew, or when the Draw, Clear, Size, DrawTo (Syntax 1), DrawToSize (Syntax 1), Print (Syntax 1), or PrintSize (Syntax 1) method is called. Even if the Image property isn't specified.

- If the Locked property is set to true, the event occurs when the Locked property is set to false to cause the barcode is redrew, or the Draw, Clear, Size, DrawTo (Syntax 1), DrawToSize (Syntax 1), Print (Syntax 1), or PrintSize (Syntax 1) method is called. Even if the Image property isn't specified.
- The event is available only for the Delphi/C++ Builder 2009 or later.

C.1.9 OnInvalidLength

Occurs when the length of the Barcode, FIMType (Only for TBarcode1D_FIM component), PatchType (Only for TBarcode1D_Patch component), Tracking (Only for TBarcode1D_OneCode component), or Routing (Only for TBarcode1D_OneCode component) property is invalid.

Syntax:

```
type
  { Defined in the pBarcode1D unit }
  TOnInvalidLength = procedure (Sender: TObject; Barcode: string) of object;

property OnInvalidLength: TOnInvalidLength;
```

Parameters:

- **Sender:** TObject; It is the object whose event handler is called.
- **Barcode:** String; The invalid value of the Barcode, FIMType (Only for TBarcode1D_FIM component), PatchType (Only for TBarcode1D_Patch component), Tracking (Only for TBarcode1D_OneCode component), or Routing (Only for TBarcode1D_OneCode component) property.

  For the TBarcode1D_OneCode component, first 20 characters are the Tracking (It is right padded with zeroes to 20 characters), then come the Routing.

Note:

- If the Locked property is set to false, the event occurs when any component property is changed to cause the barcode is redrew, or when the Draw, Clear, Size, DrawTo (Syntax 1), DrawToSize (Syntax 1), Print (Syntax 1), or PrintSize (Syntax 1) method is called. Even if the Image property isn't specified.
- If the Locked property is set to true, the event occurs when the Locked property is set to false to cause the barcode is redrew, or the Draw, Clear, Size, DrawTo (Syntax 1), DrawToSize (Syntax 1), Print (Syntax 1), or PrintSize (Syntax 1) method is called. Even if the Image property isn't specified.

1D Barcode VCL Components User Manual

6.0.1.2015 2015-09-08

Copyright © 2001-2015 Han-soft Corporation. All rights reserved.